BASIC MZ-80 SERIES



SHARP CORPORATION

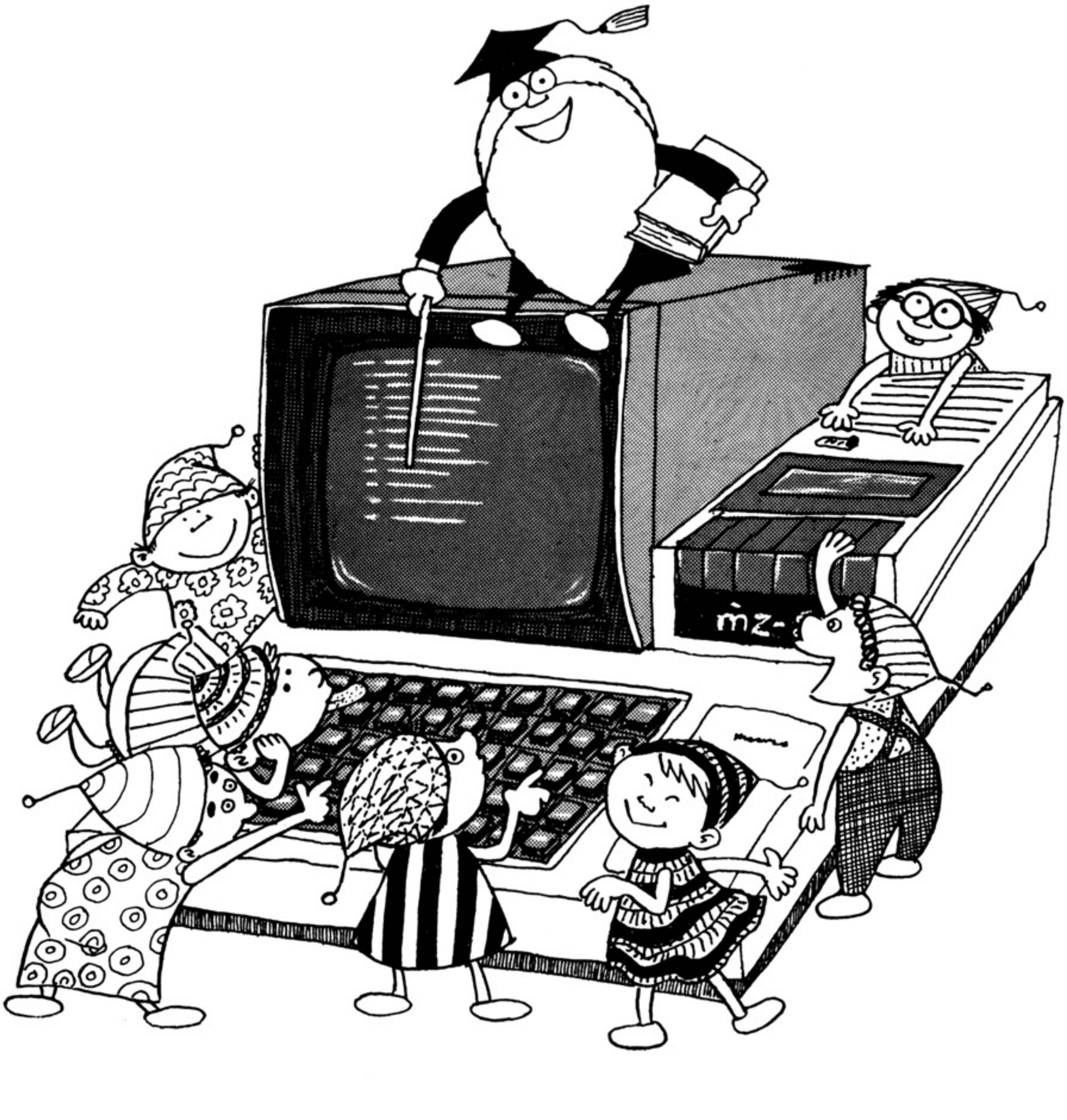
MZ-80
SERIES


Hyperboreer
Hippomolgen
Kikonen
Skylla
Trinakrie
Itaka
Kyklopen
Paphlagonier
Sparta
Ogygia
Lydien
Phönicien
Theben
Liby
Erember
Athiopen
Region des Tages
Pygmaen
Fluß
Okeanos

はじめに

このたびは、シャープクリーンコンピュータ、MZ－80シリーズをお買い上げいただき、誠にありがとうございました。
この解説書は、MZ－80シリーズのBASICについて、その使い方、文法規則を簡単な例題とともに説明しております。MZ－80シリーズクリーンコンピュータのハードウェア、あるいは、他のシステムプログラムの構成に関する事項は、それぞれのマニュアルによってください。
なお、本BASIC解説書では、BASICバージョンSP－5030に従った説明がなされております。

この「MZ－80シリーズ　BASIC解説」は、サービスお客様ご相談窓口一覧表とともに必ず保存してください。万一ご使用中にわからないことや不都合が生じたとき、きっとお役に立ちます。



ＭＺ－８０シリーズは、クリーンコンピュータを設計思想として開発されております。製品定格および命令仕様は、将来のバージョンアップで予告なしに変更・追加・削除されることがあります。そのため、提供されるモニター（ＲＯＭで提供）、システムプログラム（カセットテープまたはミニフロッピーディスケット等で提供）のバージョンナンバーには特にご注意をお願い致します。
記載事項について間違いはないものと信じておりますが、本資料は、MZ－80シリーズのBASICを御理解していただくため作製したものであり、バージョンアップによる内容の相違、誤植、記載もれなどからおこる問題点に関しては、原則的に責務は負いかねます。
尚、本書作製はモニターＳＰ-1002、BASIC ＳＰ-5030にもとづいております。

MZ-80 SERIES
BASIC解説



にんげんは昔から宇宙からの光が好きだった

だからいく時代も、日が暮れると草原に寝ころび、星降る空を眺めたものだろう。ギリシャ神話にでてくるあの若者も、そうやって宇宙海に広がる光の夢を織っていたのだ。やがて、彼は金羊毛を捜して船出した。目ざすは未来……

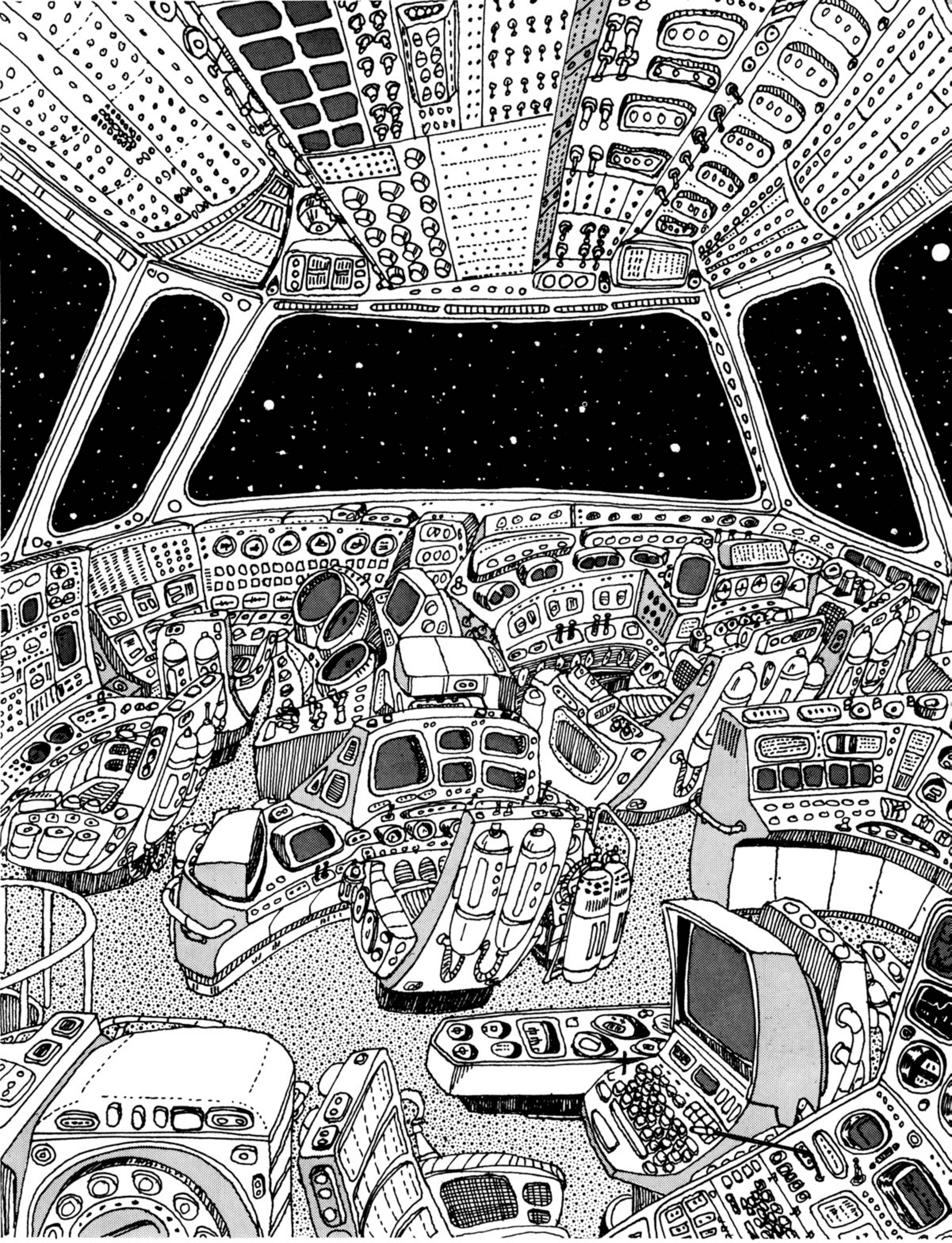
そう、南の地平線にくっきり輝くアルゴ星座はそんな星だ。そしてこの星座のシンボルは若者ののった船。勇気、未来、探究、そして憧れといったものをあらわす帆船だ。

若者の船のように、MZ－80クリーンコンピュータが船出するのはもうじきのこと……。あなたの夢とそんなふうに出会うため、シャープオリジナルのアルゴ船のマークをかたどってみました。

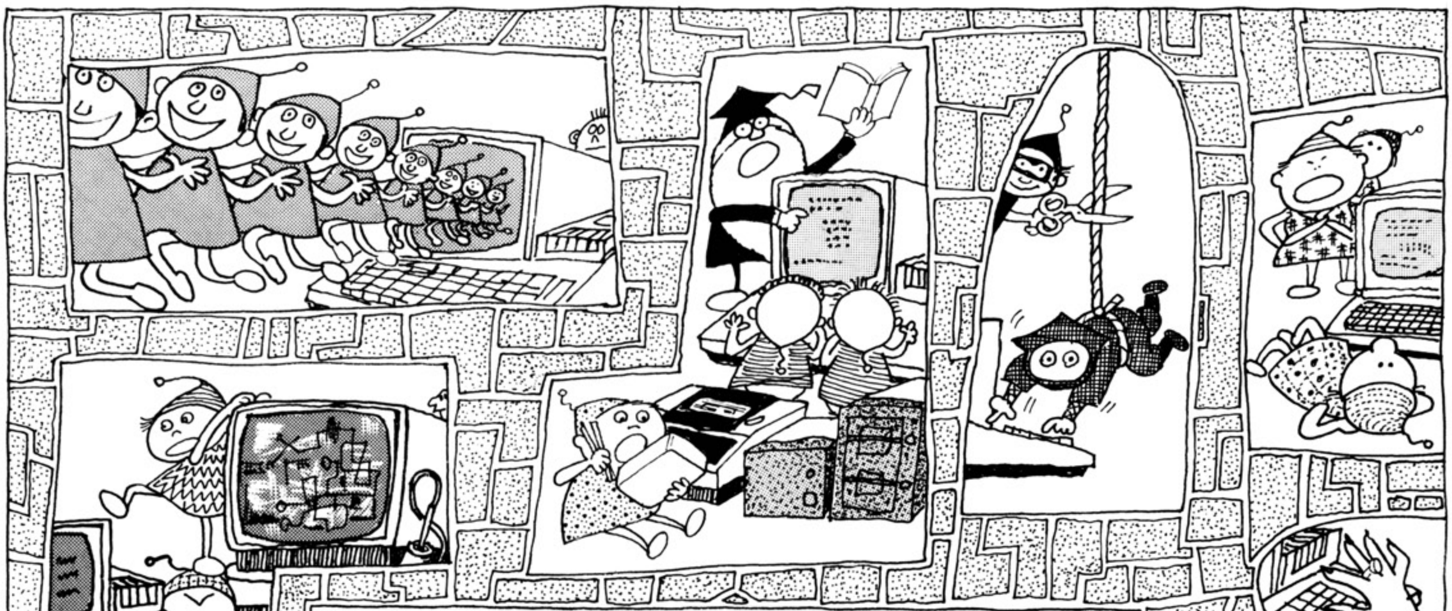
MZ-80
SERIES

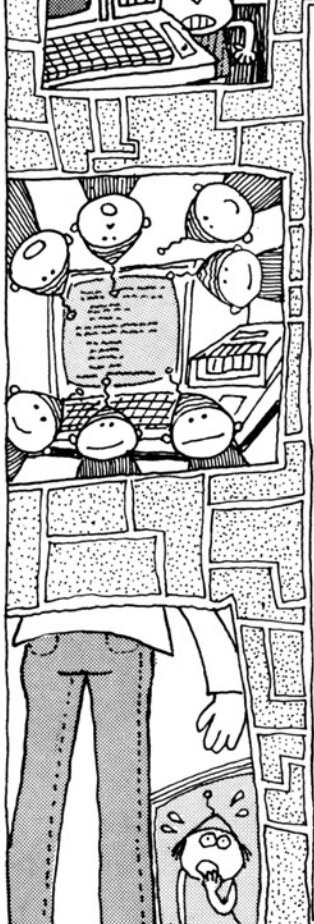


目次



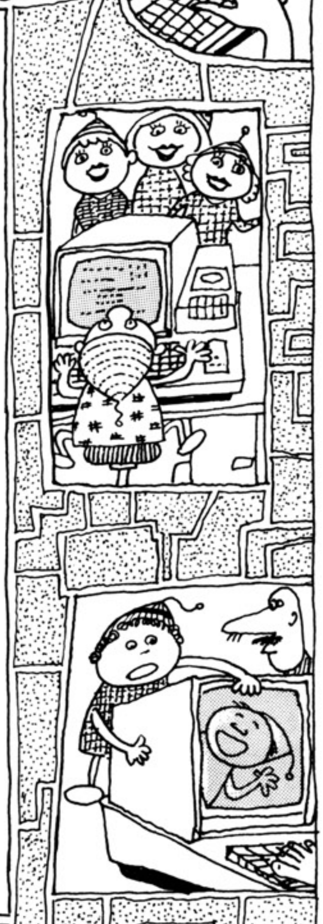
いまあなたは広大なコンピュータの世界へ旅立とうとしています。それはちょうど無限の宇宙空間への挑戦のように、未知の可能性にあふれた旅立ちにちがいありません。さあ出発です。THREE TWO ONE ZERO! LET'S GO!

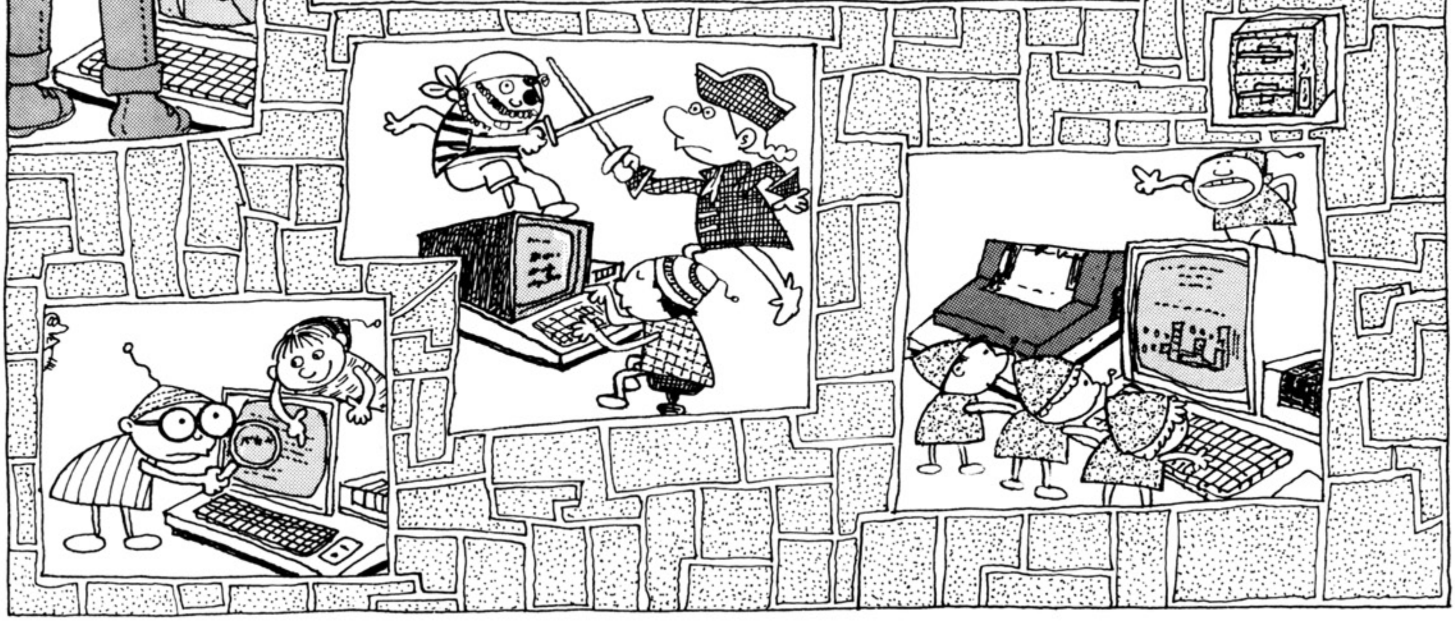




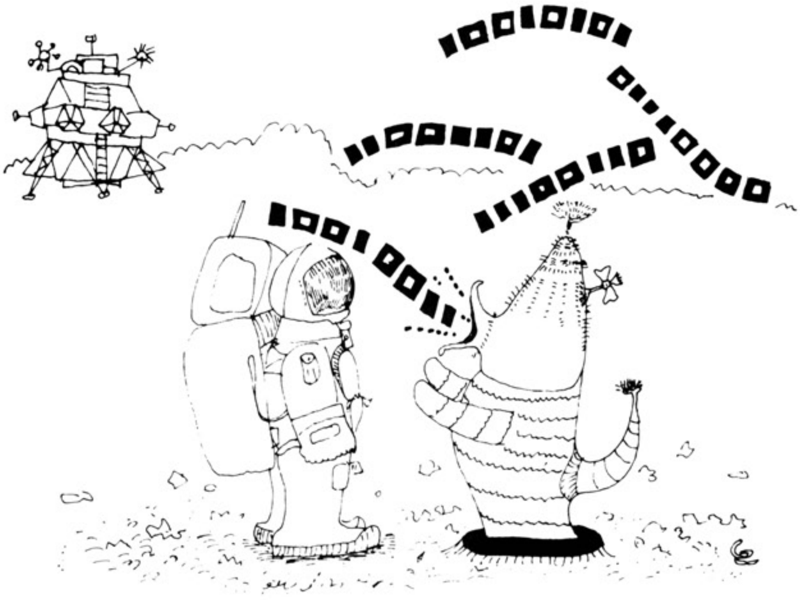
あなたへこっそり伝えたいこと

新しく何かを始めるには、まず実際に「やってみる」ことです。でも「やってみる」といっても、初めての人にとってはなかなか勇気のいることだし、第一やっぱりしんどいものです。水泳の本を読み、畳の上で練習をしたからといって、すぐに海に飛びこんだら溺れてしまいます。それは知識をたくわえても、「勘」がついていないからです。「勘」をつけるには、やっぱり基礎から勉強してスマートに「やってみる」ことです。
コンピュータはすこしぐらい使い方をまちがっても、あなたを危険な目にあわせるようなひどい道具ではありません。使いなれたら手離せないくらい便利な道具です。
この本は気楽にコンピュータを「やってみる」ことでコンピュータの本質を理解するための手引書です。コンピュータを横に置いて使いながら読んでみて下さい。

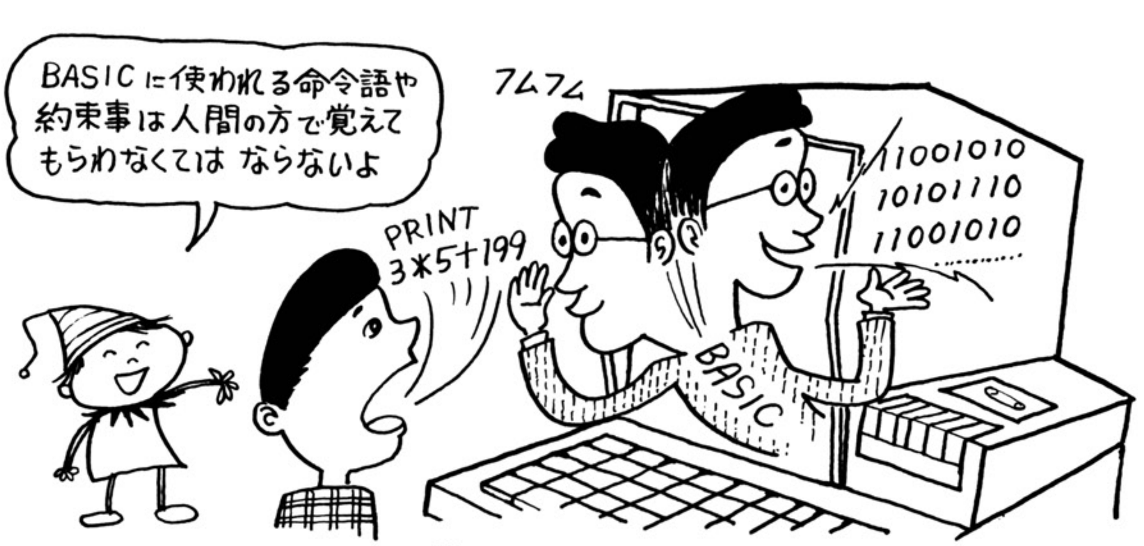




ようこそ、BASICランドへ



人間が会話をするには言葉が必要です。どんなに高い能力を持つ人達でも、お互いに通じ合う共通の言葉がなくては、力を合わせていろいろな仕事をなしとげることは不可能でしょう。
コンピュータと人間との場合も同様に、お互いに通じあう共通の言葉が必要です。共通の言葉を持つことによって、コンピュータにいろいろな命令を与えることができ、その組み合せを工夫することによっていろいろな仕事を実行することができるのです。このような言葉を「コンピュータ言語」といいます。
人間の使う言葉にも、日本語や英語やドイツ語などのように、いろいろな種類がありますが、コンピュータに使われるものにも、その用途に応じていろいろあります。マイクロコンピュータの場合、「BASIC」と呼ばれるコンピュータ言語が広く使われています。BASICはやさしい英語でできており、コンピュータがどんな仕事をすればよいかを示す一連の命令文、すなわち「プログラム」を人間とコンピュータが対話するような形で作っていくことができるのが大きな特長です。
どんなコンピュータ言語を与えてやるかは、もちろんあなたの自由ですが、これから以後当分の間は、BASICを使うことを前提としてお話を進めることにします。



BASIC

ビギナース　オール　パーパス　シンボリック　インストラクション　コード
Beginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode
初心者向きで　あらゆる目的に合う　記号を使った　命令語

コンピュータだって　BASICを暗記する！



さあ、それでは、これからあなたのコンピュータの運転開始です。まず、後ろの[POWER]のスイッチを入れてください。やがてテレビ画面に

```
**  MONITOR  SP-100X  **
*■
```

と表示が出ます。そして■がチカチカと点滅しているはずです。点滅する■をカーソルといい、キーボードのキーをたたくと、キーの文字や記号がここにあらわれ、カーソルは右側に移動して、つぎのキー入力を待つ状態になります。

この状態では、コンピュータはまだほとんど何にもできません。まだあなたとコンピュータとが互いに理解しあうための共通の言葉を覚えていないからです。できることは、このような言葉を覚え込むための能力だけです。



人間の場合、ひととおりの言葉を覚え込むためには、長い年月と努力が必要ですが、コンピュータはもちろん、ほんのわずかの操作と時間があれば事足ります。カセットテープからこのような言葉を教え込んでやることを「LOAD(ロード)する」といいます。

キーボードの中から文字を拾って

[L]→[O]→[A]→[D]→[CR]

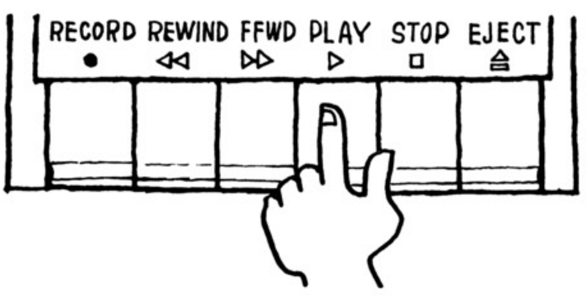
の順に、たたいてください。もし、途中でまちがえてキーをたたいてしまったら、

[INST・DEL]	をたたいて、その文字を消し改めて正しい文字を拾うか、
[CR]	をたたいて、またはじめからやり直してください。

うまくいきましたか？　正しくキー入力されていたら、下の画面のようになります。そしてカーソルの点滅は消えているはずです。

これで、コンピュータに共通言語を教え込む手はずがすっかり整いました。

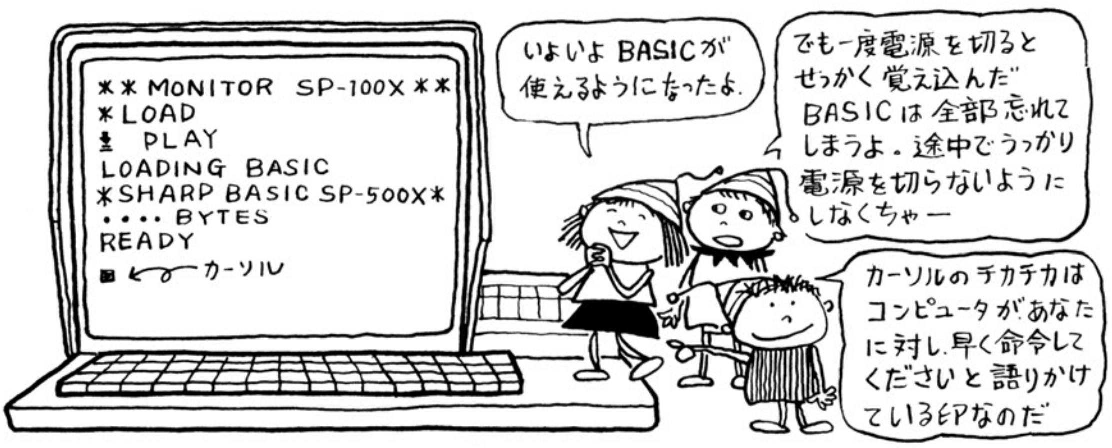
それではテープレコーダのPLAYボタンを押し、コンピュータにBASIC言語を覚え込ませてください。





BASICは　会話がお好き……





これでコンピュータはＢＡＳＩＣを学びとりました。
今度はあなたにＢＡＳＩＣを使いこなしてもらわなくてはなりません。
そうです、あなたの学習の開始です。
コンピュータを前に置き、実際に動かしながらＢＡＳＩＣを学んでください。
最初はいろいろとためしてみることが大切ですし、それが楽しく学ぶコツです。
どのキーをたたいても、決してこわれる心配はありません。キーをたたいてみて、コンピュータが何と答えてくるかためしてごらんなさい。万一途中で無限に繰り返すような動作が始まったら［ＳＨＩＦＴ］キーを押した状態で［ＢＲＥＡＫ］キーを押してください。コンピュータは仕事を中止し、命令待ちの状態にかえります。

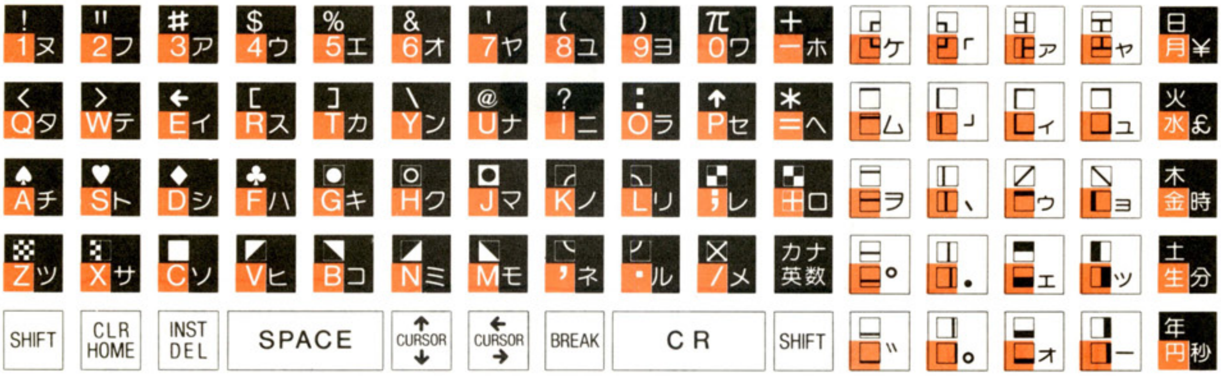
キーボードの使い方

あなたがコンピュータへ話しかける手段となるもの、それがキーボードです。それぞれのキーは色わけしてありますが、使い方によって、つぎの2つに分けられます。

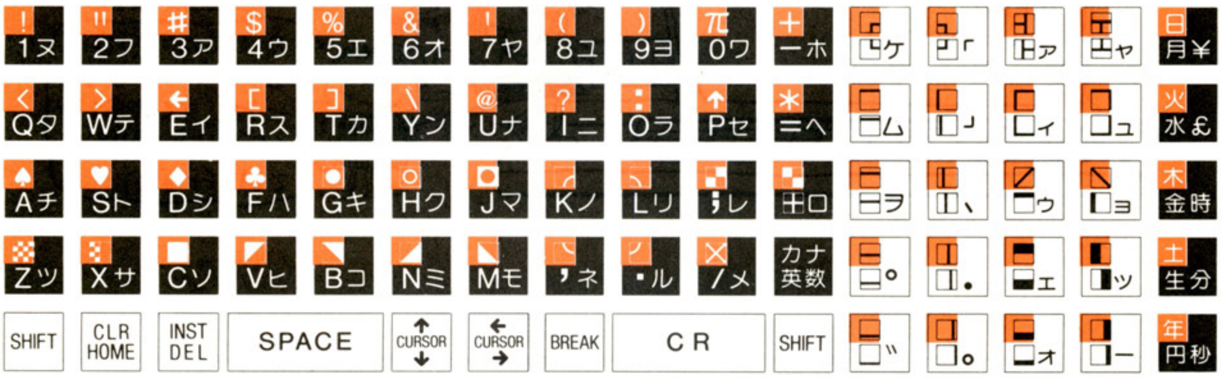
黒と青のキー……全部で204種類の文字や記号を入力するためのもので、ひとつのキーに3つずつ文字や記号があります。

黄色のキー………いろいろな制御をするための、機能的な働きをもったキーで、いちばん手前に並んでいます。

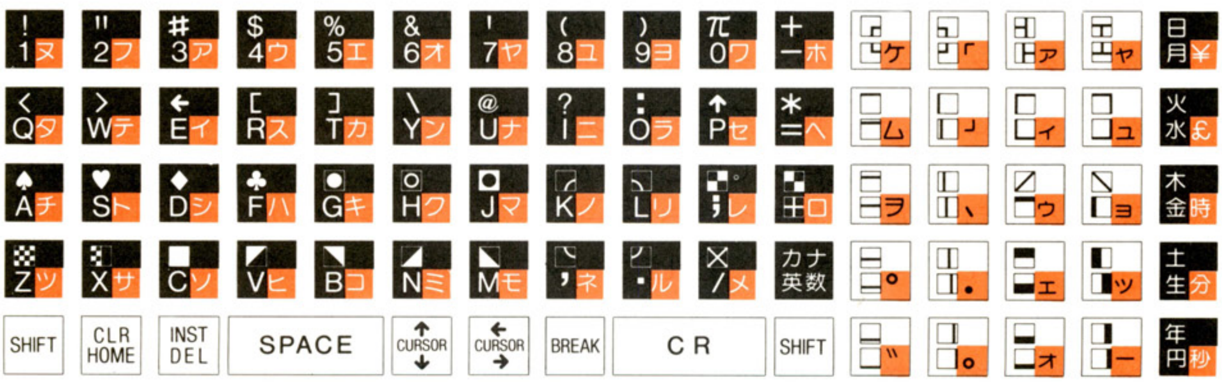
① 文字、記号のキーをたたくとテレビ画面には下図の色刷りの部分の文字や記号が表示されます。



② 特殊キーの中にある SHIFT キーを押しながら、文字、記号のキーをたたくと、下図の色刷りの部分の文字や記号が表示されます。

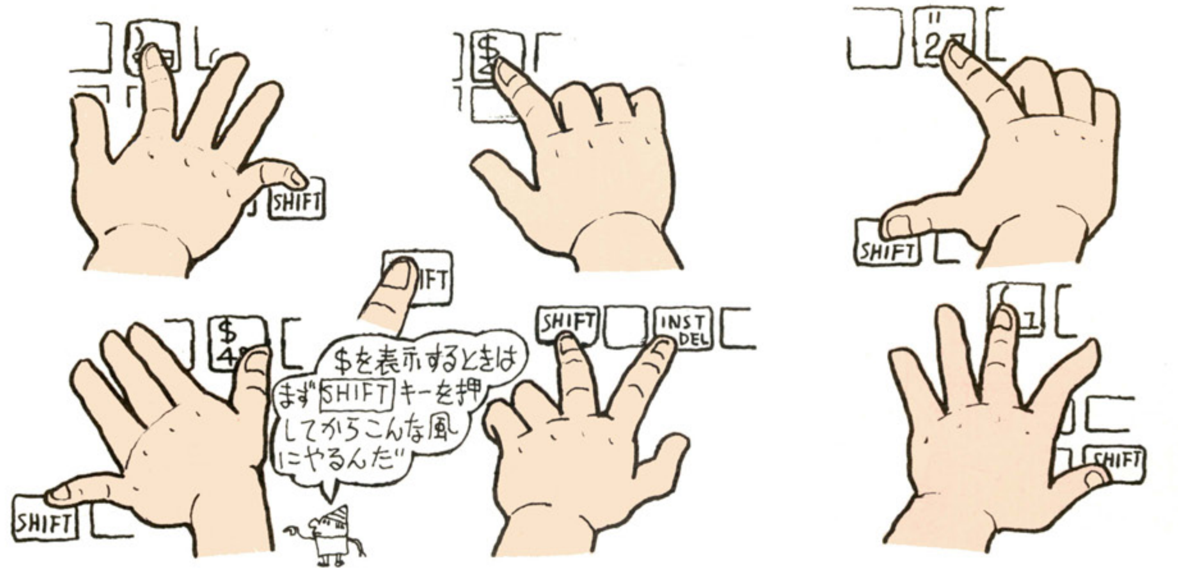


③ 特殊キーの中にある SHIFT キーを押しながら カナ・英数 キーをたたくと、今まで緑色だったランプが赤に変わります。このあとで文字、記号のキーをたたくと、下図の色刷りの部分の文字や記号が表示されます。



このあと、引き続き①に示す文字を表示する場合には カナ・英数 キーをたたいてください。ランプは緑に変わり①の状態にもどります。

黄色いキーは　魔術師



特殊キーの働き

（カーソルコントロール用特殊キーには**オートリピート機能**があり、キーを押し続けるとカーソル操作が繰り返されます。）

CR　テレビ画面に表示されている命令をコンピュータに伝えた後、カーソルがつぎの行の先頭に移ります。CR：carriage return の略。

CLR / HOME　カーソルがテレビ画面の左上すみに移ります。

INST / DEL　カーソルのすぐ左の文字を消去し、右側に続く文字を１字分左側に移動させます。空いた右端はブランクになります。DEL：deleteの略。

↑ CURSOR ↓　カーソルが下に１字分移ります。もしカーソルがテレビ画面の下端にあった場合には表示が上に１行分順送りされます。なお、カーソルが通過しても文字は消えません。

← CURSOR →　カーソルが１字分右に移ります。カーソルが右端にある場合には、１行下の左端に移ります。なお、カーソルが通過しても文字は消えません。

BREAK　コンピュータにカセットテープから入力しているとき、またカセットテープに出力しているときはその動作を中止します。

SHIFT を押しながら特殊キーをたたくと、特殊キーの働きはつぎのように変ってきます。

CLR / HOME　テレビ画面の表示を消去し、カーソルが左上すみに移ります。ただし、それによってプログラムや変数の内容は変わりません。CLR：clearの略。

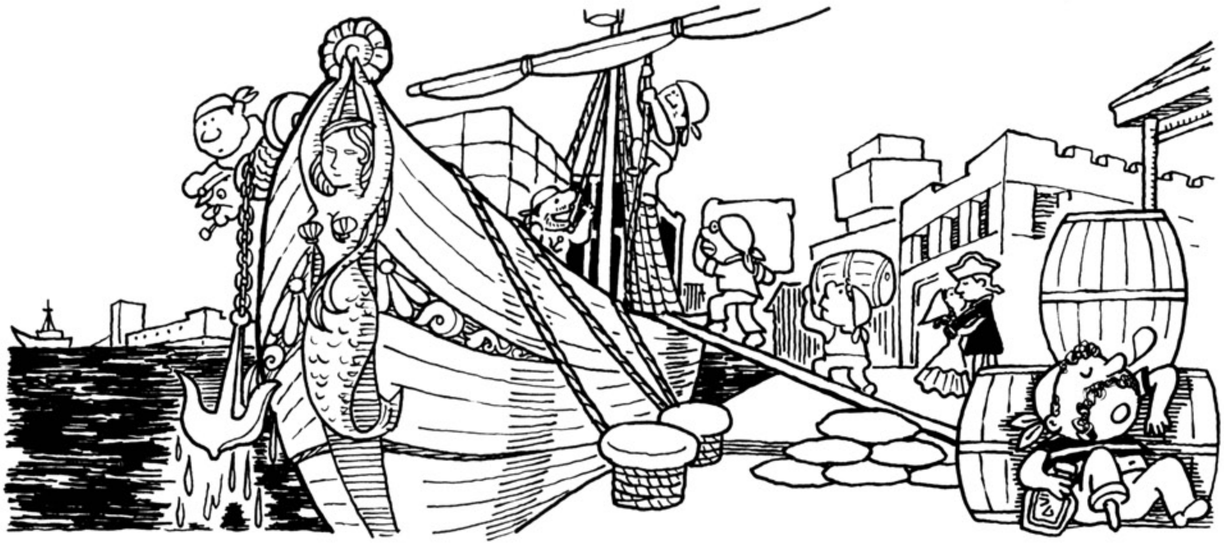
INST / DEL　カーソルの位置に空白を挿入し、右側に続く文字を１字分右側に移動させます。INST：insertの略。

↑ CURSOR ↓　カーソルが上に１字分移ります。ただし、カーソルの上端にあるときは動きません。なお、カーソルが通過しても文字は消えません。

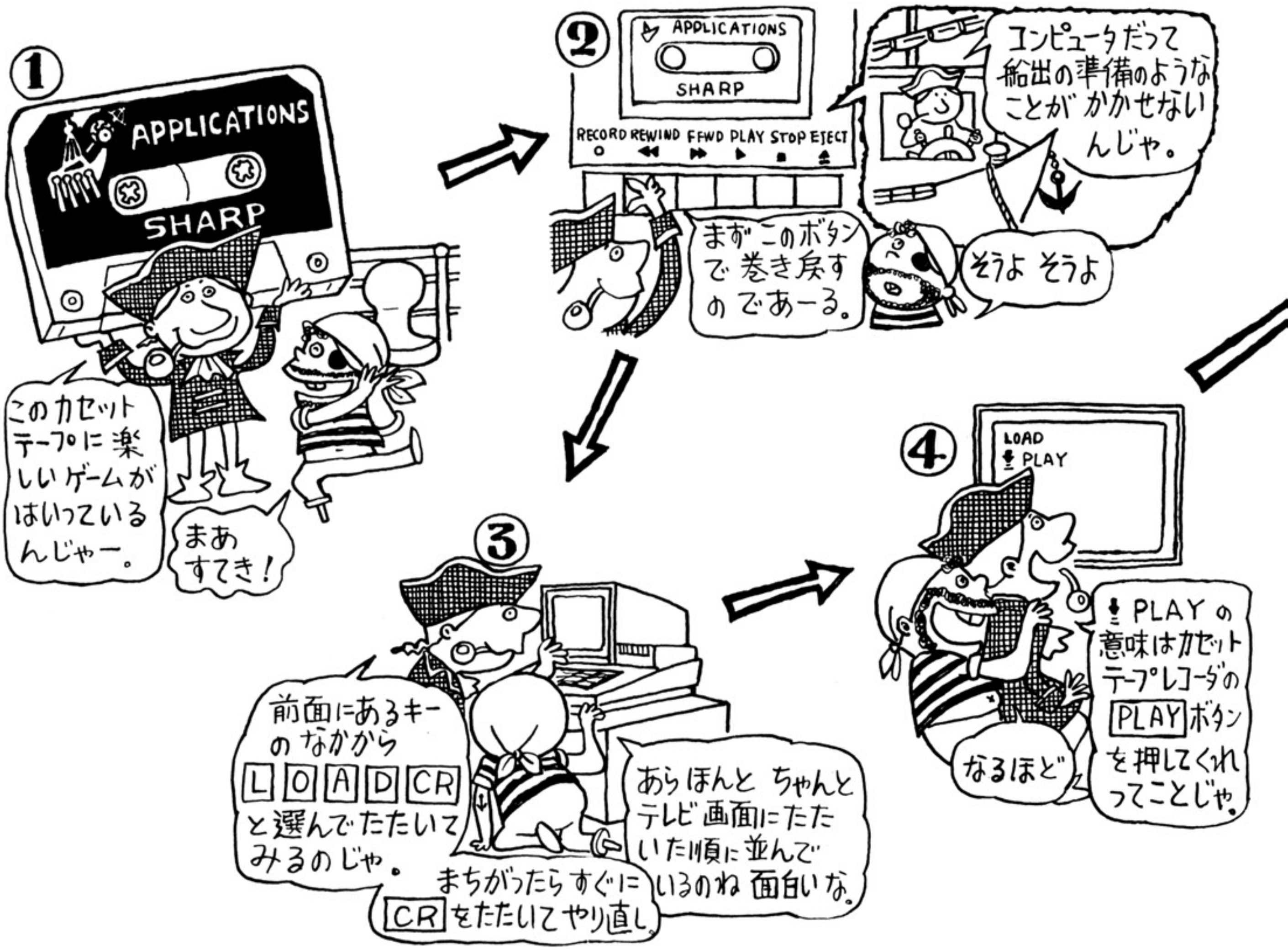
← CURSOR →　カーソルが１字分左に移ります。カーソルが左端にある場合には１行上の右端に移ります。なお、カーソルが通過しても文字は消えません。

BREAK　プログラムの実行を中止します。

ＢＡＳＩＣの海へ　いざ出帆

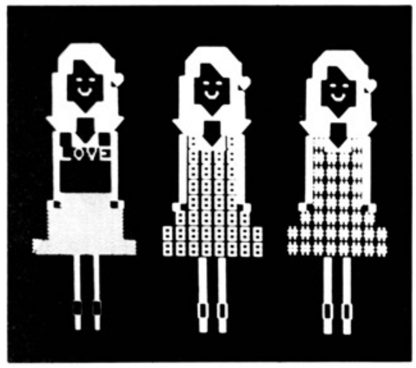


さあ楽しい航海のはじまりです。最初に私たちの乗る帆船を描いてみましょう。それはカセットテープ「ＡＰＰＬＩＣＡＴＩＯＮＳ」の中にはいっています。「ＢＡＳＩＣ」の読みこみと同様に、次の手順を追って読みこませます。たじろがずにためしてみることが一番勉強になります。
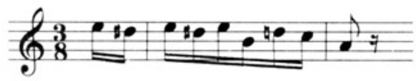


エリスのファッションショー

さて無事に出航を終えたところで、祝賀のパーティの席へ降りて行ってみることにしましょう。ちょっとした楽しいショーやゲームが行なわれていますよ……



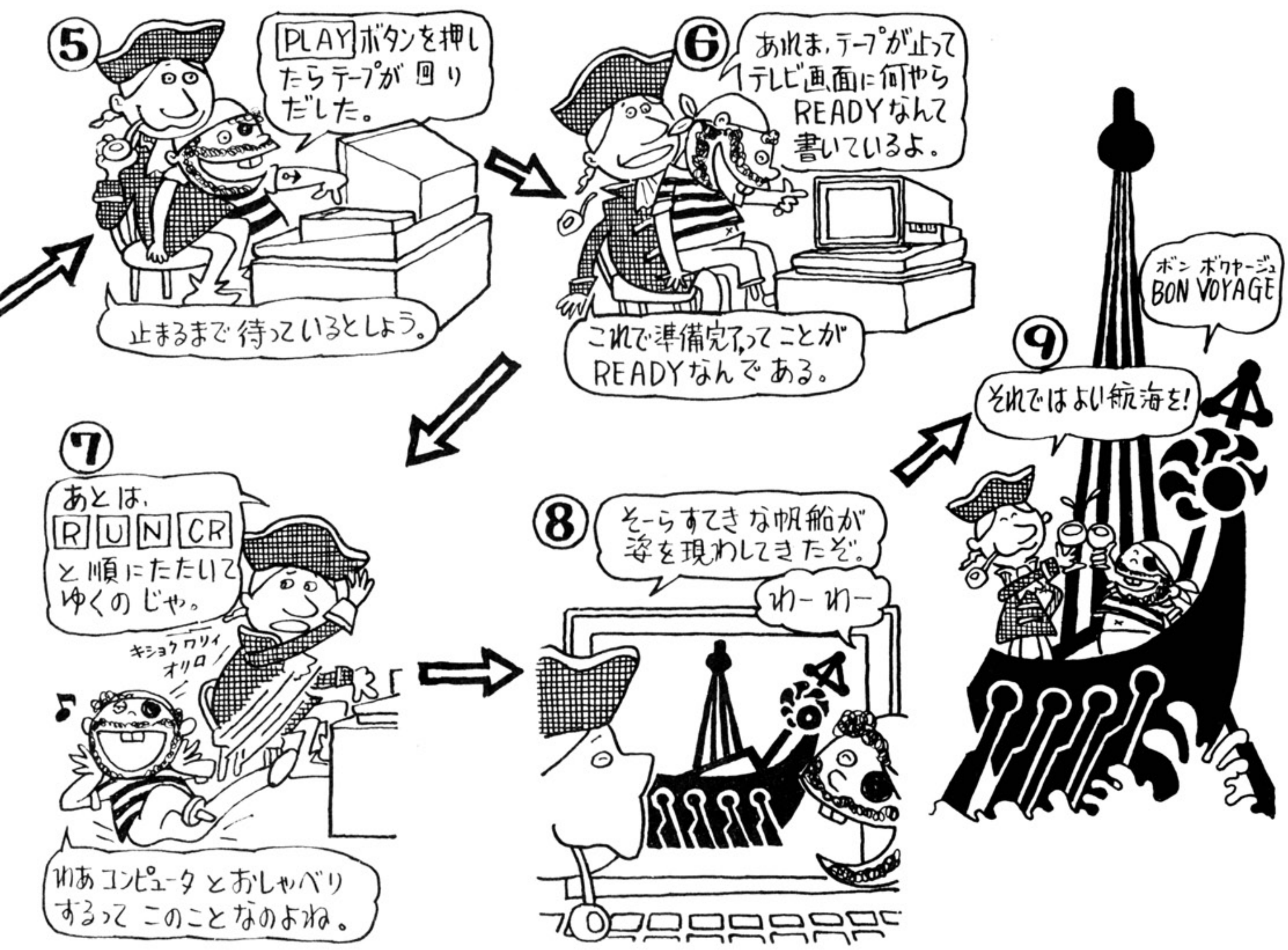
♥♥♥エリスのファッションショー♥♥♥

１０１号キャビン（船室）では、ファッションショーが行なわれているようです。そして何か音楽もきこえてきます。アルゴ号の若者たちのアイドル、エリスが登場しているのです。このショーは同じテープのA面で、船の模様のつぎに入っていますから同じように



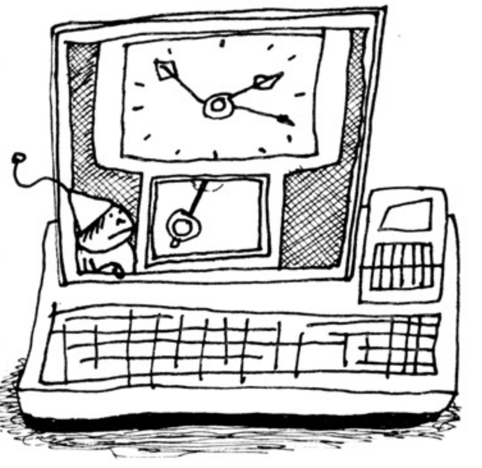
L O A D　CR

とキー入力するとテープが走り出すはずです。もしテープレコーダを STOP としておいたならコンピュータは前のように「PLAYボタンを押してくれ」と言って来ますので PLAY ボタンを押せばいいのです。LOADが終ったら、前と同じように R U N　CR としてください。おしゃれなエリスと音楽と……１０１号キャビンはとても楽しい雰囲気です。



時計の話と考えないと負けるゲームの話

コンピュータは時計を内蔵しています



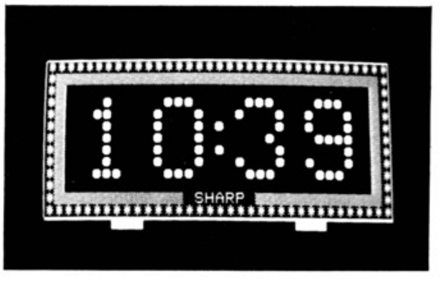
この時計は電源スイッチを入れた瞬間から「時」を刻みはじめ、その最初の瞬間に時計は自動的に

００時００分００秒

に合わせられます。「００」と書くのは意味があるのです。コンピュータ内の時計は実際には時、分、秒それぞれ２ケタがつながった６ケタの数になっているからです。
コンピュータの電源スイッチを入れたのはおよそ何分前でしたか？……ちょうど３０分前ならこの６ケタの数は、

００３０００

となっているはず。さあ、いままでと同じ手順で、つぎに記録されている「置き時計」のプログラムを LOAD してみて下さい。



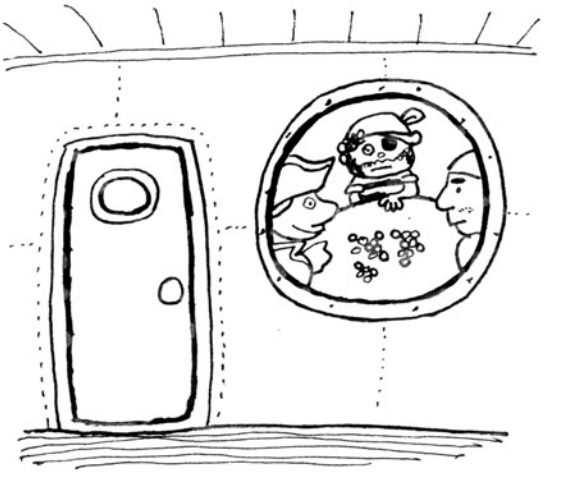
カセットテープがまた動き出し、止まったところでコンピュータは"READY"とLOADが完了したことをしらせてくれますね。そこで最初に、コンピュータの時計をあなたの時計に合わせます。たとえば、あなたの時計が現在、午前１０時３５分１２秒をさしているとして、午前１０時３６分ちょうどに合わせることにします。それはつぎのようなキー入力が必要です。

TI$="103600"

そして１０時３６分になると同時に CR キーをたたくとコンピュータ内の時計も合うわけです。

あとはもう RUN するだけのこと……

１０２号キャビンでは、気のいい船乗りたちがワイワイいいながら山くずしゲームに興じています。さっそくこのプログラムを LOAD することにしましょう。どうするって？……手順はいままでと全く同じこと。このゲームのルールはつぎの２つだけ。



- ♣ あなたとコンピュータが交替に、どれかひとつの山から１個以上の石を取っていきます。
- ♣ 山がだんだん少なくなって、最後に残った石を取った方が勝ち。

……そしてプログラムの世界へ

プログラムのLOADは終わりましたか？……
さあ、ここで [R][U][N] と打ってゲーム開始です。

[R][U][N] [CR] ⇦ [CR] をたたくのをお忘れなく

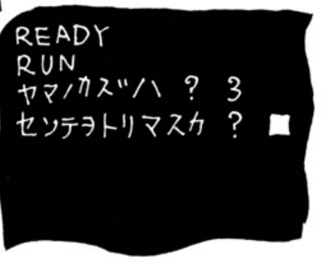
これからさきはあなたが考える番です。考えないと負けるゲームへご招待！

1



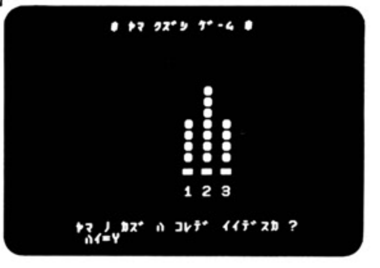
はじめに山の数をいくつにするかコンピュータがききます。
たとえば3つにするには
[3] [CR]
と打って行きます。山の数は1から8まで指定できます。

2



つぎに先手・後手をきいてきます。
先手をとるには [Y] [CR]
後手をとるには [N] [CR]
を打ちます。
いま [Y][CR] と打って先手をとりましょう。

3



いよいよゲーム開始！
3つの山が表示されました。
そして、
ドノヤマニシマスカ？■
ときいてきます。たとえば、
[2][CR] とします。つぎに
イクツトリマスカ？■
ときいてきたら取る石の数をキー入力します。たとえば、[5][CR] とすると、第2の山から石を5個とることになります。

• •

いままで見てきたコンピュータの働きは何によるのだろう？カセットテープの中にあるものは何？……これこそあなたの乗りだした海。**BASICプログラムの世界**なのです。ゲームにはゲームのための、時計には時計のためのプログラムが作成されているのです。ここでプログラムというものがどんなものか、ちょっとのぞいてみます。キーをつぎのようにたたいてください。

[L][I][S][T][CR]

画面につぎつぎ現われたのが山くずしゲームのプログラムです。いまあなたの船はゆっくりと舳先をめぐらし、進路をまっすぐ目的地へ向けたところです。
そう！……目指すのは自分でプログラムを作成したり、直したりできるようになること。



• •

ダイレクトモードって何だろう

もしお望みならコンピュータを電卓のように使うこともできるのです。このような動かし方を「ダイレクトモード」ともいいます。ためしに電卓と同じように

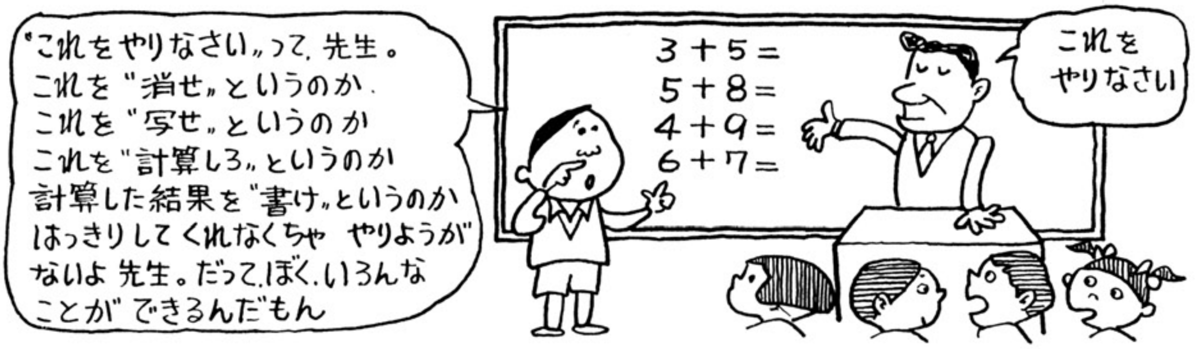
5+8=

とキー入力してみてください。+は SHIFT キーを押しながら + キーをたたけばできましたね。

でもコンピュータは、テレビ画面にはキー入力したとおりの文字しか表示せず、もちろん CR キーをたたいても計算は実行してくれません。

あなたのコンピュータと電卓とは、そこが全然異なっているのです。コンピュータに必要なことは、5+8=をどうしろというのか、という命令なのです。





PRINT

コンピュータを電卓と同じように使うためには、5+8の計算結果をテレビ画面に表示させることが必要です。そのための命令としてPRINT命令というのがあります。これを使ってつぎのような順序でキーをたたいて命令を伝えてみましょう。

P R I N T 5 + 8 CR

テレビ画面には、キーをたたくにつれて、つぎのような文字が表示されていくはずですが、あなたの場合はいかがですか?

READY……………………さあ、どうぞという意味
PRINT5+8……………………5+8の結果を表示しなさい、そして
命令のおわりを示す CR キーをたたくと
13……………………これがPRINT命令を実行した結果
READY……………………おつぎは何をやりますか
■……………………カーソル

四則演算もバッチリ

PRINT命令を使って、もう少しいろいろな計算をやってみませんか？…… もっとけた数の多い演算とか、何回も足し算を続ける計算とか、そして引き算も…………。
もし、あなたが掛け算や割り算まで手を拡げたくなったら、コンピュータは、普通の数学記号とは、ちよっと違った記号を使っていることを知っておいてください。

掛け算の記号………＊	アスタリスクと呼ばれています。
割り算の記号………／	スラッシュと呼ばれています。

つまり

3×5　のことをコンピュータでは　3＊5
6÷2　　　　〃　　　　　　　　6／2
でなくてはならないのです。

(　　)つき計算

コンピュータは、普通の電卓の計算よりはるかに複雑な計算式も平気でやってのけます。かっこつき計算がそれです。ただし数学記号の場合

3×6×〔6+3×｛9−2×（4−2）+1｝〕

のように、かっこの種類を違えて書きますが、コンピュータの場合すべて一種類、(　)だけで間に合わせています。

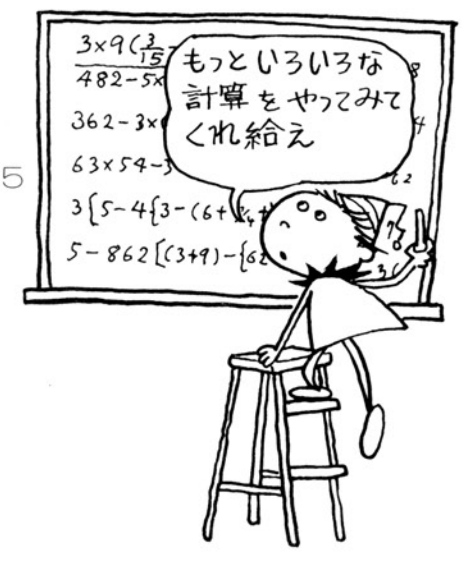
3＊6＊（6+3＊（9−2＊（4−2）+1））

このようにしても、コンピュータは、内側のかっこ内の計算を先にやらなければならない、というルールを忘れてしまったり、間違えたりはしないからです。



練習問題

```
PRINT (6+4)/(6-4)
 5
PRINT 3*(5+9*(9-2)-6/(4-2))+5
 200
PRINT (3+4)*(5+6)
 77
PRINT (10+20)/6*(2+3)
 25
PRINT (10+20)/(6*(2+3))
 1
```



ストリング？　計算式？

```
PRINT 3+5
```

そして CR をたたくと8になりますね。では、ここでクォーテーションマーク「"」で計算式を囲んでみてください。

```
PRINT"3+5"  そして CR
3+5
```

おや？……違った結果が出てきましたね。もうひとつだけためしてみましょう。

```
PRINT"HELLO MY FRIEND" CR
HELLO MY FRIEND
```

"　"でくくった文字や記号が、そっくりそのままテレビ画面に写し出されるしくみになっているのです。このクォーテーションマークの中にある文字や数字の列のことを「ストリング」といいます。Stringとは、糸とか、一連、一列などという意味です。

PRINT"3+5"

これがストリング
クォーテーションマークでくくられています

PRINT 3+5

これは計算式
ストリングではありません。

ストリングについては、まだまだ多くのことを知っていただく必要があります。そしてそれらを自由に使いこなすことにより、コンピュータのおもしろさが倍増するのです。
でも、それはもっとあとの、お楽しみ……。



PRINT ???

今後、かなり数多くお付き合いしなくてはならない命令語です。そのたびにPRINTと入力するのは面倒とお考えの方は

　PRINT　の代りに　?

をたたいてください。コンピュータの中では、?を自動的にPRINTと読みかえてくれます。

```
?3*5
 15
?(3+4)*10
 70
```



プリントの第1接近と第2接近とは

PRINT命令のあとには、ストリングや計算式などの複数個の項目をつけてやることができます。そして、そのとき各項目は、セミコロンやコンマで分離してやらなくてはなりません。

```
PRINT"3+5=";3+5 CR
3+5=  8
```

↑セミコロン

" "の中の計算式はストリングです。実際の計算は、セミコロンのあとに続く計算式によってなされているわけですね。ですから" "の中が100+100であろうと0+0であろうと、コンピュータは何の矛盾も感じません。ためしてみてください。

では「;」の代りに「,」を使ったらどうなるでしょうか?

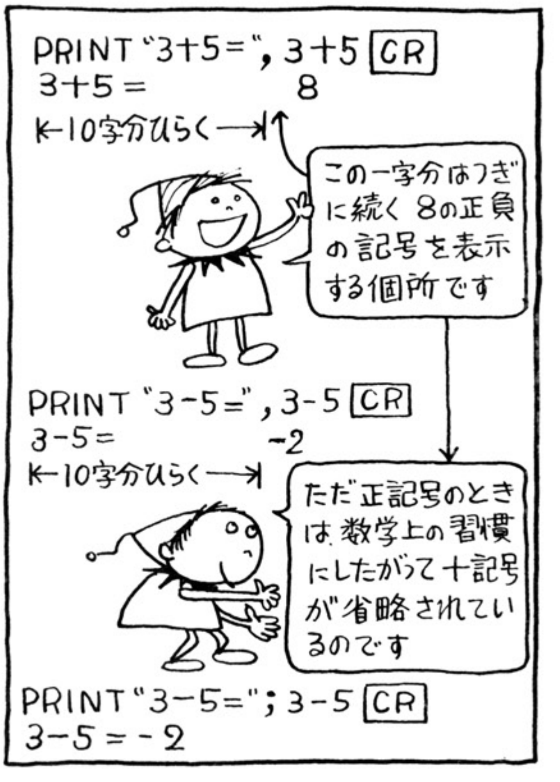
```
PRINT"3+5",3+5 CR
3+5            8
```

↑コンマ

おやおや!…… 3+5の結果が、ぐんと離れて表示されました。つまりつぎのようなちがいがあるのです。

; ……**すぐとなり合わせに表示される。**

, ……**右のように10字分離れたところから表示される。**



コンマで区分した場合、ストリングの終りから6字分ひらくのではなく、ストリングの始めからかぞえて10字分ひらくことに、ご注意ください。つまりストリングの長さには関係なく、8という数字が表示されるのです。

```
PRINT"12345",3+5 CR
12345          8
←──10字分──→
PRINT"123456789",3+5 CR
123456789      8
←──10字分──→
```

もしストリングが10字以上の場合には、8は自動的にさらに10字分離れてしまいます。

```
PRINT"123456789012",3+5 CR
123456789012                  8
←──────20字分──────→
```



第2接近



第1接近

走れコンピュータ

いよいよ、何行にもわたった命令文のプログラムとの出会いです。

```
10  A=3
20  B=5
30  C=A+B
40  PRINT  A,B,C
50  END
```



とくに説明の必要がない程のプログラムですね。

C=A+B

の計算式でA=3、B=5の場合を計算し、A、B、Cをテレビ画面に表示しなさい、という意味のプログラムです。
各文の先頭に書かれた数字を文番号といい、コンピュータは必ず文番号の小さい値から、大きい値へと、順序正しく実行していきます。したがって、あとで上のプログラムに、たとえば

35　D=B−C

などと、途中に新しい命令文を追加することも可能です。コンピュータは、文番号の順にプログラムを実行していきます。したがって、文番号は上の例のように10行飛びにしておき、あとで必要に応じて命令文の追加ができるようにしておくのが賢明なやり方です。なお、文番号は1から65,535まで自由に選ぶことができます。

文番号　命令文　CR



文字をテレビ画面に表示しただけでは、コンピュータに入力したことにはなりません。各命令文を表示し終ったら、そのつど CR キーをたたくことによってはじめてその文番号内の命令文をおぼえこませたことになります。

そこでたとえば

35　CR

とすると、文番号35の命令文はコンピュータから消えてしまいます。



さあ、それではプログラムを実行させてみましょう。
RUN CR とたたいてください。

```
RUN
  3          5          8
```

←――10字分――→←――10字分――→

正負の記号が入るための1字分。負の値のときにマイナス「−」が表示されます。



ＬＩＳＴではやわかり

いろいろと試行錯誤を繰り返しながらコンピュータとの対話を続けていると、いつの間にやら、始めに入力したプログラムが画面から消えてなくなっていることがありますね。
でも、心配はいりません。コンピュータは、プログラムを決して忘れているわけではないのですから。もう一度先に入力したプログラムを見たいときは、

　ＬＩＳＴ CR

とキー入力してください。コンピュータが記憶しているプログラムを全部テレビ画面に表示してくれます。もし、何十～何百行にもわたるプログラムで、一度に全部を表示できないときは、プログラムの一部分を表示させることも可能です。



ＬＩＳＴ　－３０ CR	文番号３０までのプログラムを表示。
ＬＩＳＴ　３０－ CR	文番号３０以降のプログラムを表示。
ＬＩＳＴ　３０－５０ CR	文番号３０～５０のプログラムを表示。
ＬＩＳＴ　３０ CR	文番号３０のプログラムを表示。

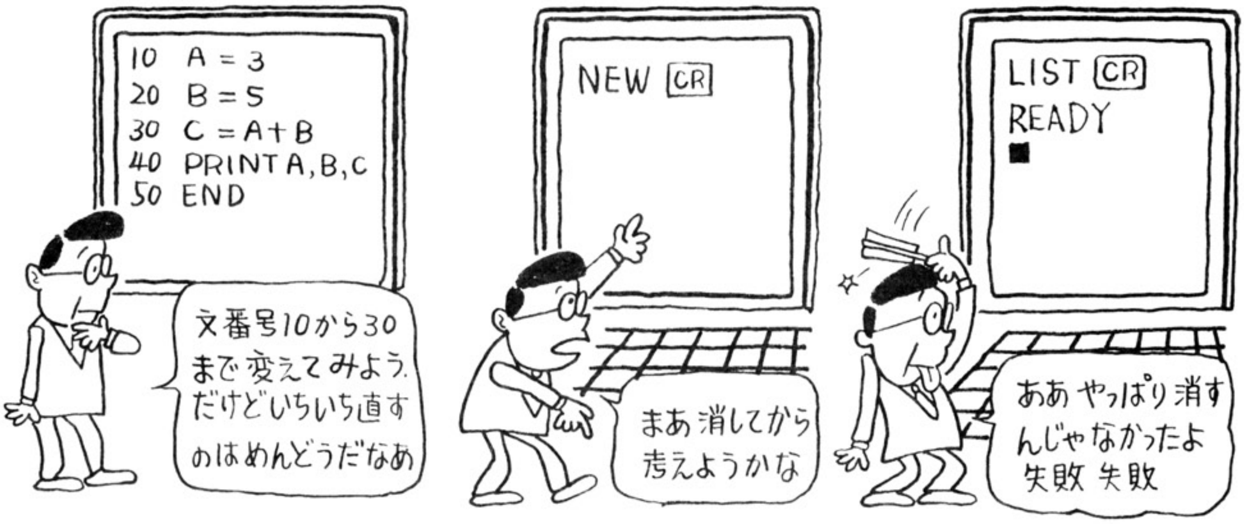
ＮＥＷの結末は……

新しくプログラムを入れるときは、前のプログラムをＮＥＷで消してしまいましょう。そうでないと、２つのプログラムが重なって、ゴチャゴチャになってしまうことがあるからです。

　NEW CR

によって、すべてを消去することができるのです。念のためにＬＩＳＴを出して確かめてみてください。





エラーいこっちゃなあ……

```
10  A=3
20  B=5
30  C=A+B
40  PRINT  A,B,C
50  END
```



24ページのプログラムと同じものですね。うまく走ってくれましたか？
もし途中の命令文に誤りがあると、コンピュータはそのことをあなたに教えてくれます。たとえば

```
50  EMD
```

と、NとMとをまちがえていると、文番号40までは命令どおり実行してくれますが、コンピュータにとってEMDとは何のことやらわからず、

```
SYNTAX  ERROR  IN  50
```

あなたにSYNTAX　ERROR、つまり文章構成上のエラーを教えてくれます。さっそく

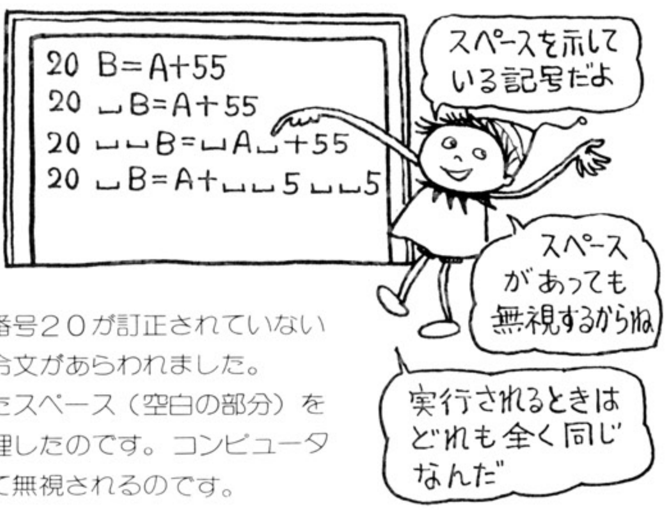
```
50  END
```

と打ち直してください。コンピュータは、同じ文番号の命令文がふたつあるときは、後から入力した命令文の方を取り上げるのです。
これでプログラムは完全になりましたか？　もしそうならためしに、たとえば文番号20の命令文を、わざとまちがえてみませんか？

```
20  5=B
```

これで文番号20の命令文は訂正されているはず。でも、果してそうでしょうか？…………念のためにLISTしてみてください。

```
10  A=3
20  B=5
30  C=A+B
40  PRINT  A,B,C
50  END
205  =B
```



アレアレ、おかしなことになりました。文番号20が訂正されていないだけでなく、文番号205という奇妙な命令文があらわれました。
実は、コンピュータは20と5の間にあったスペース（空白の部分）を無視して、ひとつながりの文番号として整理したのです。コンピュータにとってスペースは全く無意味なものとして無視されるのです。

デートの場所は？

コンピュータを使うと、筆算や電卓ではとてもやり切れないような面倒な計算も別に苦にはなりません。ですから、普段思い付いても面倒さが先に立って試してみたことのなかったような、しかも変わったテーマについて、計算してみるのも一興です。たとえば、もし「のみ」が人間や象と同じくらいの大きさになって、その跳躍力もそれに比例して大きくなったらどれ位飛べるだろうか、とか、もし人口が現在の比率でいつまでも増え続けたとしたら……等々いろいろあります。これらの計算をためていけば、あなたはちょっとしたもの知り博士になれるにちがいありません。

浅草の観音様と、奈良の大仏様がデートをしようと思い付き、同時に立上って相手の方向に歩き出しました。お二方の出合いの場はどの辺になるでしょうかという問題なんかいかが？

東京浅草寺の聖観世音菩薩は約5.5㎝の立像、東大寺の大仏様は盧舎那仏で約16.2ｍ、立上ったら恐らく34ｍはあるでしょう。なお、鎌倉の大仏様は高徳院清浄泉寺にあり、阿弥陀如来の像で約11.4ｍ、立ち上ったら24ｍでしょうか。また、小さな浅草の観音様の名誉のために申し添えると、観音様をお守りする仁王様は大仏様に負けず劣らず大男のはず。何しろ仁王様のはきものは長さ4.5ｍ、幅1.5ｍ、重さ400㎏のジャンボワラジなのですから。

浅草ー東大寺の直線距離は約380㎞、浅草ー鎌倉が約49㎞です。普通の人が時速４㎞で歩くとし、そのスピードは背の高さに比例すると考えると、さあもう計算できますね。ぜひ試してみてください。

	推定身長	歩く速さ	東大寺ー浅草寺を歩く時間	高徳院ー浅草寺を歩く時間
浅草の観音様	0.055 m	0.13 km	2923 時間	377 時間
奈良の大仏様	34	80	4.75	0.61
鎌倉の大仏様	24	56.5	6.73	0.87
人間	1.7	4	95	12.25

4.5mのわらじをはく仁王様の身長は約30mぐらいはあるでしょう.



命令文を直す?

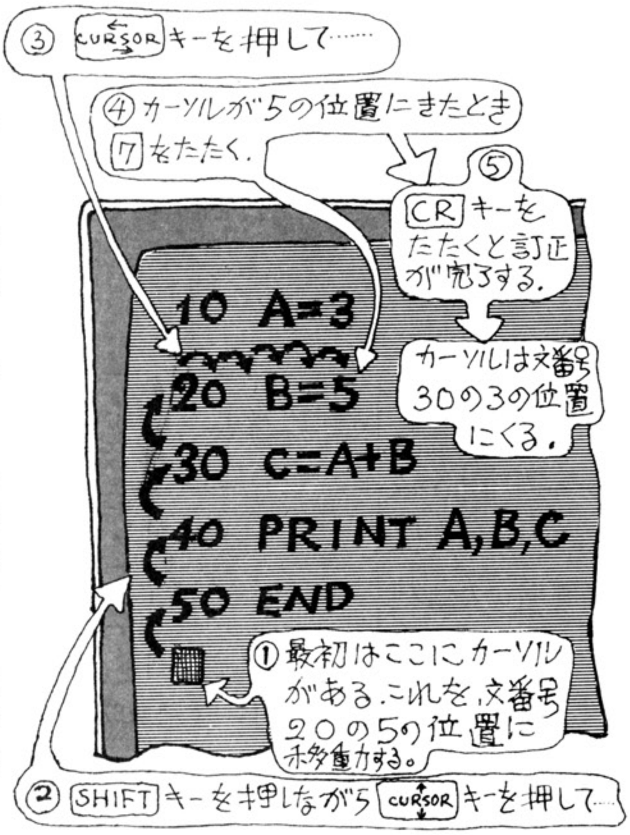
さて、コンピュータと対話しながら作っているあなたのプログラムに関して、

間違いを修正したいとき
修正して、よりよい命令文にしたいとき
修正して、別の命令文にしたいとき
すでに完成している命令文から、ほんの一部を修正して新しい
命令文を作り出したいとき
…………

このようなときの命令文修正や追加削除の方法を、研究してみましょう。

カーソルの移動

命令文内の文字を訂正するためには、その文字の位置にカーソルを移動させなくてはなりません。カーソルの移動キー(上下左右)および、INST(挿入)、DEL(削除)それにスペースキーは、**オートリピート機能**があり、キーを押し続けると繰り返して実行します。(SP−5030)
では文番号20のB=5の5を7に変えてみましょう。動かす手順は右の図を見てください。



これでテレビ画面上のプログラムは修正できました。しかし、これだけではまだコンピュータが記憶しているプログラムを修正しているわけではありません。記憶内容を変更させるためには、必ず CR キーをたたいてやる必要があります。
エ?…… その前に7と入れるべき所を6と入れてしまったのですか?つまりカーソルのすぐ左隣の文字を修正するわけですね。それには二つの方法があります。

その1 INST・DEL キーをたたく方法。
カーソルはそのまま一字分左に移って、すぐ左隣にあった文字が消えてしまいます。ここで再び 7 キーをたたけばよいのです。もちろんこのあと CR キーをたたく必要があることはいうまでもありません。

その2 CURSOR キーを使って左方向に移動する方法。

SHIFT キーを押しながら ←CURSOR→ をたたいてみてください。カーソルは左方向に押している間移動し続けます。ここでふたたび 7 キーをたたきます。

命令文を直す！

文字の追加

26ページのプログラムで、文番号30の命令文をつぎのように変更したいときは、

30　D=100+A+B　← CR キーはまだたたかない

カーソルをAの上に移動してください。そしてつぎのような手順でキー操作してみてください。

SHIFT キーを押しながら
INST・DEL キーを押す。

ちょうど**100+**を追加できる4字分のスペースを作ります。それでは、このスペースの部分に、**100+**を入れてください。CをDに修正する方法は、もう説明を要しませんね。
せっかくここまで修正したのですから、文番号を30から35に修正した上で CR キーをたたき、文番号40の命令文を

40　PRINT　A，B，C，D

に直してRUNさせてみたらいかがでしょう。

```
RUN
  3          5          8          108
```

文字の削除

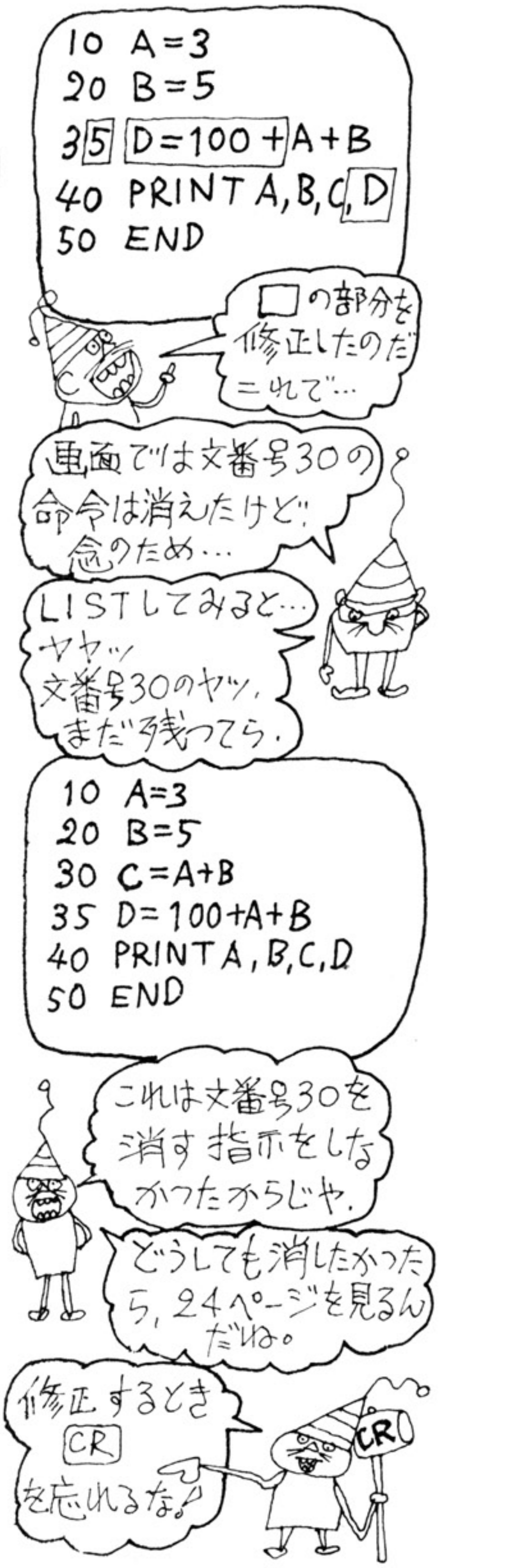
35　D=100+A+B

この命令文を直してみましょう。たとえば

35　D=A+B+C

に修正したいときは、Aの上にカーソルを移動させ、 INST・DEL キーを押してください。
A+Bの部分が=印のすぐとなりまで移動します。
+Cの追加はもう説明を要しないと思います。

```
RUN
  3          5          8          16
```



コンマとセミコロンの大研究

おさらいのために、ふたたび26ページのプログラムを例にとります。

```
10  A=3
20  B=5
30  C=A+B
40  PRINT  A, B, C
50  END
```

```
  3                  5                  8
←──── 10字分 ────→←──── 10字分 ────→
```

数字の一字前に正負の記号のためのスペース

でしたね。つまり文番号40でA、B、Cの間にコンマを使うと、10字ずつ離れて数値が表示されています。
では新しくいくつかの命令文を追加したプログラムを作り、RUNしてみましょう。追加する命令文はつぎの4個です。

```
32  D=B↑A
34  E=B*A
36  F=B/A
45  PRINT  D, E, F
```

```
RUN
  3               5               8
  125             15              1. 6666667
```

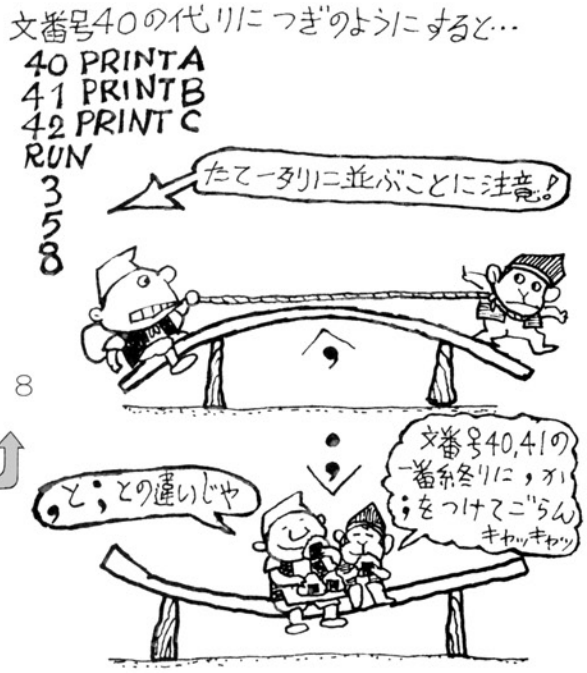
文番号40、45の「,」をセミコロン「;」に修正してもう一度走らせてみましょう。修正には、一度LISTしたあとカーソルをうまく使ってできるだけスマートにね。

```
RUN
  3  5  8
  125  15  1. 6666667
```

正負の記号のためのスペース

「;」は、表示する文字や記号を、ぴったりとくっつけてしまう性質があります。念のために文番号40の命令文の終りにも「;」を追加してRUNしてみてください。

```
40  PRINT  A;B;C;
RUN
  3  5  8  125  15  1. 6666667
```







コロンだほうがいいこともある

コロンの使い方

```
10  A=3
20  B=5
30  C=A+B
32  D=B↑A
34  E=B*A
36  F=B/A
40  PRINT  A;B;C
45  PRINT  D, E, F
50  END
```



このプログラムを見ると、各行とも短かい命令文の集まりでできています。しかし、もしこの程度の長さのプログラムなら、そしてもしあなたがそうしたいと望まれるなら、ひとつの文番号の中ですべてを処理することも可能です。

```
100A=3:B=5:C=A+B:D=B↑A:E=B*A:F=B/A:PRINT
A;B;C:PRINT  D, E, F:END
RUN100
```

コロン「:」はひとつの文番号の中に、ふたつ以上の命令文を入れ込むときに使う記号です。このような命令文を多重命令（マルチステートメント）といいます。ひとつの文番号の中には、２行にわたる文を書き込むことができます。１行は、４０文字ですから、文番号を含め７６字までの範囲なら、自由に使うことが可能なわけです。



あといくつ?……SIZE

コンピュータにプログラムをつぎつぎに覚え込ませていると、あなたが自由に使える記憶容量があとどれくらい残っているか知りたくなるものです。このようなとき

```
PRINT  SIZE
```

としてやると、コンピュータは、残りの記憶容量の数（バイト数）を教えてくれます。ほぼ、この数値に当たる文字数の命令文を記憶することができるという意味です。



A＝BとB＝Aは同じなの？

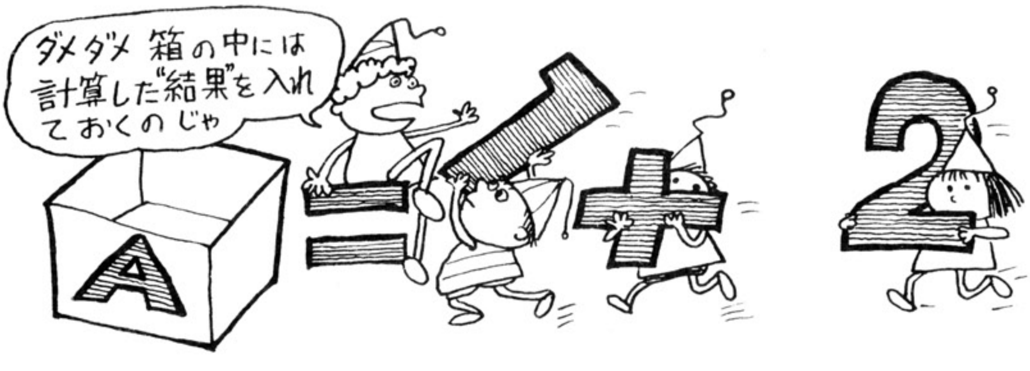
今まで何気なく使ってきた「＝」記号に注目してみましょう。まずつぎのプログラムを実行させてみて下さい。

```
10  A=1
20  PRINT  A,
30  A=A+2
40  PRINT  A
50  END
RUN
 1          3
```



１と３が表示されました。文番号３０を見てみると、A＝A＋２とあります。もしこれが等式だとしたら、両辺からAをひいて０＝２となり矛盾します。そうではないのです。

記号「＝」の意味は右辺にある計算式などの結果を左辺に用意されたAという名前の箱へ代入するのです。



文番号１０で１という値がAという名前の箱へ代入され、文番号３０の右辺で箱Aの中の値と２が加算されて、記号「＝」によって箱Aに代入されます。その時箱Aにはじめ入っていた１という値はもう箱の中にはありません。
つぎの２つのプログラムのちがいも、記号「＝」が代入命令であることによってはじめて理解できるでしょう。

```
10  A=5
20  B=7
30  PRINT  A, B
40  A=B
50  PRINT  A, B
60  END
RUN
 5          7
 7          7
```



```
10  A=5
20  B=7
30  PRINT  A, B
40  B=A
50  PRINT  A, B
60  END
RUN
 5          7
 5          5
```

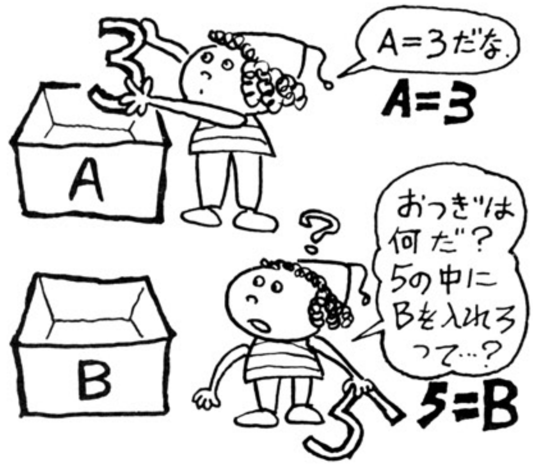


コンピュータの大好きな変数は………

コンピュータの命令文の中で使われる変数は、数学上の使い方とは、ちょっと違った考え方をしなくてはなりません。
命令文に使われる変数は、数値を収容するための箱があり、その箱に付けられた名称とでも言うべきものなのです。そして

　B＝5

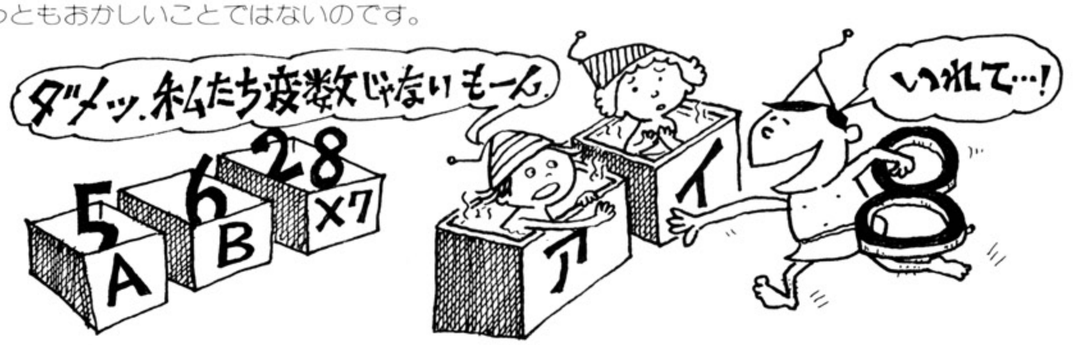
というのは、Bという箱に5という数値を入れなさい、という意味なのです。したがって、26ページに示したような使い方は、たとえ文番号とまちがえたりしないとしてもコンピュータにとって、はなはだ困った命令文ということになってしまいます。何しろ5の中にBという箱を入れなさい、というのですから。




```
10　A＝3
20　B＝5
30　A＝A＋B
40　PRINTA
50　END
RUN
　8
```



数学的な定義から見たら、このプログラムには明らかに矛盾がありますが、コンピュータにとってはちっともおかしいことではないのです。



コンピュータの変数になりたがる文字

1．2文字以内の文字の組み合わせでなくてはなりません。3文字以上にわたる変数は、命令文としては記憶されますが、コンピュータ処理にあたっては、3文字以後の文字は一切無視されてしまいます。たとえばABC、ABDは表示するときは区別されますが、演算処理のときは、同じ変数ABとみなされてしまうのです。

2．変数として使える文字はつぎの通りです。

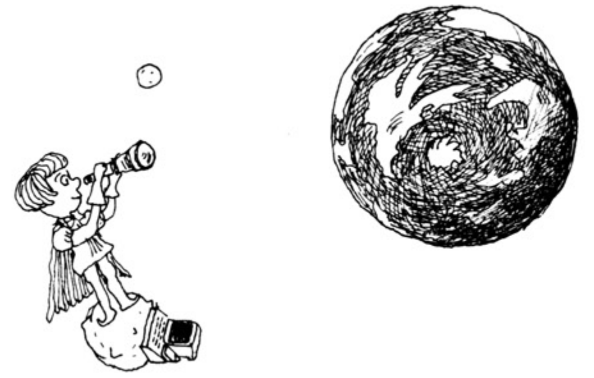
(1)A～Z　アルファベット　26とおり
　例：A，M，Z

(2)アルファベットに1けたの数字（0～9）を付加した文字　260とおり
　例：A0，K5，Z9

(3)アルファベットの2文字を組み合せた文字　ただしBASICステートメントにあるIF、ON、TOなどは使えません。

地球を計算する

星の王子様の地球観測…「地球というのは、太陽系の中にあるあの青い惑星だ。少しゆがんでいるが、直径は約1万3千キロメーター、軌道計算からすると、質量はだいたい6×10^{17}万トンあるようだぞ」
星の王子様は、さっそく自分のコンピュータのもとに走り地球の体積VE、表面積SE、それに平均密度ZEを計算してみようと、こんなプログラムを作ってみました。



```
10  DE=13000                    ←変数DEに地球の直径を代入するんだ
20  WE=6E+17                    ←変数WEに地球の質量を代入するんだ。
30  SE=4*π*(DE/2)↑2            ←これで変数SEに表面積が入るぞ
40  VE=4*π*(DE/2)↑3/3          ←これは変数VEに体積を代入するものだ
50  ZE=WE/VE*(1E−2)             ←これで変数ZEに密度が代入できるんだ
60  PRINT"チキュウノチョッケイ・・・";DE;"キロメートル"
70  PRINT"チキュウノヒョウメンセキ・";SE;"ヘイホウキロメートル"
80  PRINT"チキュウノタイセキ・・・・";VE;"リッポ゜ウキロメートル"
90  PRINT"チキュウノシツリョウ・・・";WE;"マントン"
100  PRINT"チキュウノミツト゛・・・・";ZE;"キロク゛ラム/リッポ゜ウメートル"
110  END
```

星の王子様は、地球の大きさが少しわかった気がしました。出てくる単位に注意して下さいね。
ここで、*とか+とか↑などがひとつの計算式の中にあるとき、どの順で計算してゆくのか注意しておきましょう。演算の優先順序をつぎに示します。



次に示す式はちょっと複雑な組み合わせになりますがどうです、ちがいがわかりますか?……

```
┌2+3↑2=11              ┌12/3*2=8
└(2+3)↑2=25            └12/(3*2)=2

┌2*2↑3=16              ┌12/3↑2=1.3333333
└(2*2)↑3=64.000001     └(12/3)↑2=16
```

天才アルキメデスと謎の兵隊

三角形の内角の和は１８０°？ アルキメデスはそうひらめくとすぐ、道端にすわりこんで三角形を描いてみるのでした。そこへ謎の兵隊が現れて槍をつきつけました。

兵隊　「アルキメデスよ、お前の命もこれまでだ、覚悟せい！」

アルキメデス　「ちょっと待て、これを計算するから」

兵隊　「何だって？　角度Ａが３０°、Ｂが直角で、角度Ｃの大きさを出すのか。６０°に決ってらあ、それに辺ＣＡの長さがわかっていれば、辺ＡＢとＢＣの長さも、この三角の面積もすぐ出るよ」

アルキメデス　「馬鹿な」

兵隊　「ＢＡＳＩＣでプログラムすればいいのさ。ＣＡ＝１２としてみるか」

```
１０　Ａ＝３０：Ｂ＝９０：ＣＡ＝１２
２０　ＡＢ＝ＣＡ＊ＣＯＳ（Ａ＊π／１８０）
３０　ＢＣ＝ＣＡ＊ＳＩＮ（Ａ＊π／１８０）
４０　Ｓ＝ＡＢ＊ＢＣ／２
５０　Ｃ＝１８０－Ａ－Ｂ
６０　ＰＲＩＮＴ"ＡＢ＝"；ＡＢ，"ＢＣ＝"；ＢＣ，"ＣＡ＝"；ＣＡ
７０　ＰＲＩＮＴ"メンセキＳ＝"；Ｓ
８０　ＰＲＩＮＴ"Ａ＝"；Ａ，"Ｂ＝"；Ｂ，"Ｃ＝"；Ｃ
９０　ＥＮＤ
```



角度Ｃの大きさを辺ＡＢとＢＣの長さを使って逆正接関数ＡＴＮで出してみましょう。それには

```
５０　Ｃ＝ＡＴＮ（ＡＢ／ＢＣ）＊１８０／π
```

とすれば度の単位で出てきます。同じ答が得られましたか。

関数家の人々　個性いろいろ

ＳＩＮ（Ｘ）などの関数の仲間をもう少し紹介しましょう。それぞれカッコを背おっていますがどの関数も、カッコ内には定数や変数、または数式のいずれも入ることができます。

関数	ＢＡＳＩＣ記号	計算する値	例
整数化	ＩＮＴ（Ｘ）	Ｘをこえない最大の整数	ＩＮＴ（3.14）＝3 ＩＮＴ（0.55）＝0 ＩＮＴ（−7.9）＝−8
絶対値	ＡＢＳ（Ｘ）	Ｘの絶対値	ＡＢＳ（2.9）＝2.9 ＡＢＳ（−5.5）＝5.5
符号	ＳＧＮ（Ｘ）	Ｘ＞０ならば１ Ｘ＝０ならば０ Ｘ＜０ならば−１	ＳＧＮ（５００）＝１ ＳＧＮ（０）＝０ ＳＧＮ（−3.3）＝−１
指数関数	ＥＸＰ（Ｘ）	e^x （e＝2.7182818）	ＥＸＰ（１）＝2.7182818 ＥＸＰ（０）＝１
常用対数	ＬＯＧ（Ｘ）	$\log_{10} X$ ただしＸ＞０	ＬＯＧ（３）＝0.47712126
自然対数	ＬＮ（Ｘ）	$\log_e X$ ただしＸ＞０	ＬＮ（３）＝1.0986123
平方根	ＳＱＲ（Ｘ）	$\sqrt{X}$ ただしＸ≧０	ＳＱＲ（９）＝３ ＳＱＲ（０）＝０





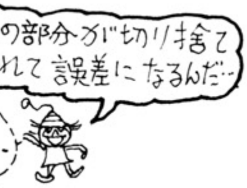


ＰＲＩＮＴ　２＊２　と　ＰＲＩＮＴ　２↑２　は同じ？

おや？……２↑２では 4.0000001 とおかしな端数がついてきましたね。これは計算式としては確かに正しくても、計算がどうしても有限桁で行なわれるため、思わぬ誤差がはいりこんだのです。

たとえば２↑２は級数展開と呼ばれる公式を使って、

$$2\uparrow 2=1+\frac{2\ln 2}{1!}+\frac{(2\ln 2)^2}{2!}+\cdots+\frac{(2\ln 2)^n}{n!}+\cdots\cdots$$



これではコンピュータの悲鳴が聞こえるようです。コンピュータに誤差はつきもの。それでも、私たちの生活にはあまり気にもならない誤差ではないかな？

自由な関数定義……DEF FN

今までいろいろな関数をみてきましたが、ここでそれらの関数を組み合わせたものを、新しい関数として定義するDEF　FN命令を説明します。いくつかの定義の例を並べてみましょう。

DEF　FNA（X）＝2＊X↑2＋3＊X＋1　……$2X^2+3X+1$をFNA（X）と定義します。

DEF　FNB（X）＝SIN（X）↑2＋COS（X）↑2……$\sin^2 X+\cos^2 X$をFNB(X)と定義します。FNB(X)はいつも1になりますね。

DEF　FNE（V）＝1／2＊M＊V↑2　……$\frac{1}{2}MV^2$ をFNE（V）と定義します。

DEFは「定義する」（define）という意味です。新しい関数はFNを頭につけて命名します。カッコの中のXとかVを「引数」と呼びます。
たとえば3番目の関数（運動エネルギーのようですね）を使ったら、

```
10  DEF  FNE(V)=1/2*M*V↑2
20  M=5.5:V=3.5
30  PRINT  FNE(V),FNE(V*2),FNE(V*3)
40  END
```

という具合に初速度がVのときの運動エネルギー、速度が2倍3倍と大きくなったときのエネルギーが表示されます。特に長いプログラムで何度か同じ形の関数を用いるときにDEF　FN命令の便利さは無類です。

高度1万メートルからの転落！
1万メートルから墜落するときの速度と、高度は1秒ごとにどんな変化になると思いますか？
プログラム中の関数FNV（T）は時間T経過したときの落下速度の関数、FNH（T）はそのときの高度を表わす関数というわけです。
重力加速度Gと空気抵抗の係数Kと、落ちはじめた高度Hを文番号20で指定してあります。





```
10  ?"C":T=0
20  G=9.8:K=0.15:H=10000
30  DEF  FNV(T)=G/K*(1-EXP(-1*K*T))
40  DEF  FNH(T)=H-G/K*(T+1/K*EXP(-1*K*T)-1/K)
50  ?"H"
60  ?"ジカン";T:MUSIC"□A0"  ← ピッとなる命令です。88ページで説明します。
70  ?"ソクド";FNV(T)
80  ?"コウド";FNH(T)
90  T=T+1:GOTO  50← これは文番号50へプログラムを移す命令で、41ページで説明します。
```

こちらはINPUT、返事がほしい!

コンピュータに変数の値を知らせるのに、いままでは

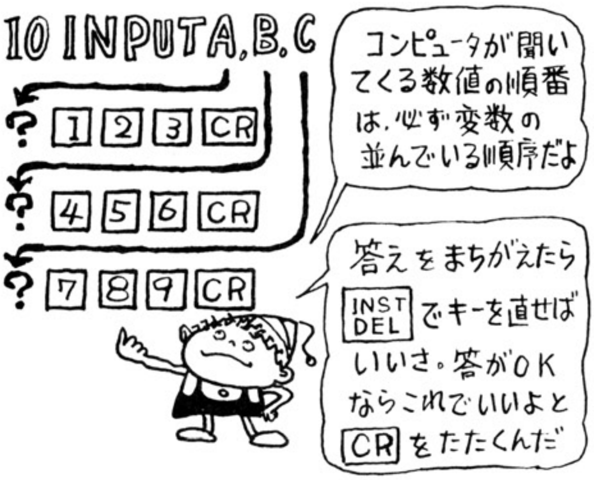
```
10  A=3
20  B=5
………………
```

のように、はじめに数値をきめてしまう方法をとっていました。
コンピュータに変数の値を知らせるのに、いくつかの方法があります。INPUTという命令を使うのも、そのひとつ。




```
10  INPUT  A, B, C
20  D=A+B+C
30  PRINT  A, B, C, D
40  END
RUN
?  ■
```

← カーソル



見なれない表示が出てきましたね。
?■は、コンピュータがあなたに対し、INPUT命令に続く一番目の変数Aの数値を問い合わせてきているのです。あなたはこの問いかけに答えて数値を入れ、これでよいと CR をたたいてください。
アレレ、また同じ表示が出ましたね。これは2番目の変数Bの値です。もし変数が3個であれば、コンピュータは3回きいてきます。もし途中でまちがえて、0~9以外のキーで答えて CR をたたくと

```
DATA  ERROR
```

と表示し、また最初から数値の問い合わせをはじめます。

```
10  INPUT  A, B, C, D
20  INPUT  E, F, G, H
30  PRINT  H, G, F, E
40  PRINT  D, C, B, A
50  END
```

ストリング

```
10  INPUT  "A=?"; A
20  INPUT  "B=?"; B
30  INPUT  "C=?"; C
40  S=A+B+C
50  M=S/3
60  PRINT "ゴウケイハ"; S, "ヘイキンハ"
    ; M
70  END
```

セミコロン



プロポーズの返事はイエスかノーか

ある晴れた日曜日、シャレた感じの喫茶店で向かいあってる彼氏と彼女。彼氏のほうは43歳でスポーツマンタイプだが一見プレイボーイふうです。また彼女は美しい22歳のお嬢さんで、なかなか賢こそうに見えます。

彼氏「君には一目ぼれなんだ、どうだい、結婚してくれないか」
彼女「ええ、そんなにまでおっしゃるなら年の差なんて気にしないわ。でも、いますぐはいや。わたしの年があなたの年の半分になるまで待っててね」

彼氏の年をA、彼女の年をB、そして彼女が何年か待ってというからX年待つことにします。X年後には、彼氏はA+X歳になり、彼女はB+X歳になります。彼女の年はそのとき彼氏の年の半分になるべきだから、A+X=2(B+X)という条件が必要です。これをXについて解くと

X=A−2B

となりますね。

```
10 PRINT"カレシ ノ トシハ"
20 INPUT A
30 PRINT"カノジョ ノ トシハ"
40 INPUT B
50 X=A−2*B
60 PRINT"アト ";X;" ネン マッテ!"
70 END
RUN
カレシ ノ トシ ハ
? 43 [CR]
カノジョ ノ トシ ハ
? 22 [CR]
アト −1 ネン マッテ!
```



ああ、−1年待つなんて不可能ですね。つまり1年前なら結婚できたはずですが……。さて、コンピュータにいきなり、?と質問されるとハテ何の変数だったかな?と困ってしまうことがありますね。そこでこのプログラムでは、文番号10と30に問い合わせの内容を示すストリングを入れてみました。そして文番号60では答えのストリングも。
このプログラムのINPUTの方法についてはもうすこし簡単にできます。文番号10と30をつぎのように直し、文番号20と40のプログラムは削除してみてください。

```
10 INPUT"カレシ ノ トシハ ? ";A
30 INPUT"カノジョ ノ トシハ ? ";B
```

(↑ セミコロン)

文番号50以下は上のプログラムと同じです。

```
RUN
カレシ ノ トシ ハ ? 43 [CR]
カノジョ ノ トシ ハ ? 22 [CR]
アト −1 ネン マッテ!
```



DATAさん、お手をどうぞ、READより

変数を知らせる、もう一つの方法です。

```
10  READ  A, B, C, D
20  X=A+B+C+D
30  PRINT  X
40  DATA  3, 5, 7, 9
RUN
  24
```

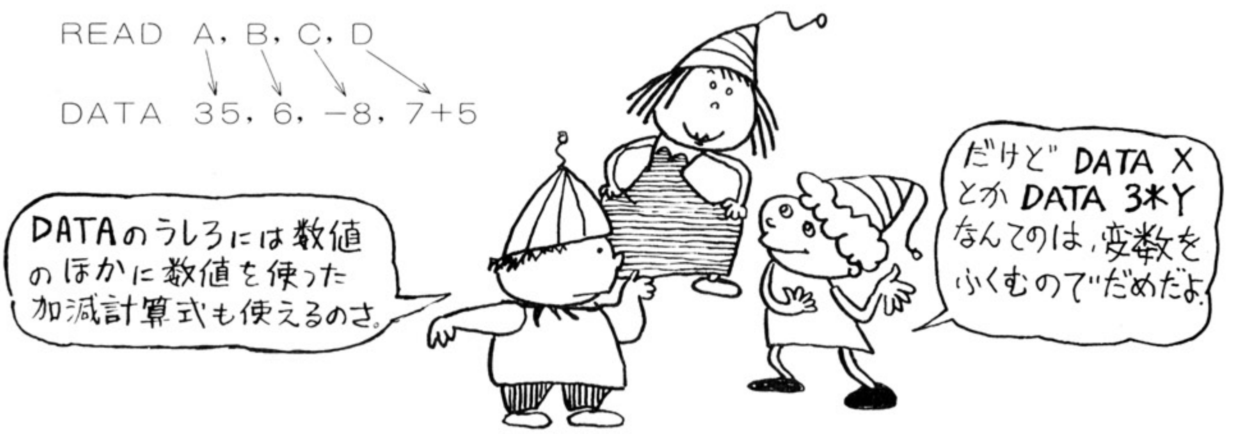


この命令文は、あとに示す内容の中から数値をひろい出して計算を進める方法です。
これには、READ、DATAの二種類の命令文が使われます。

READ　A，B，C，D	……………………いくつかの変数を並べる
DATA　10，11，12，13	………READのあとに続く変数と同じ個数の数値

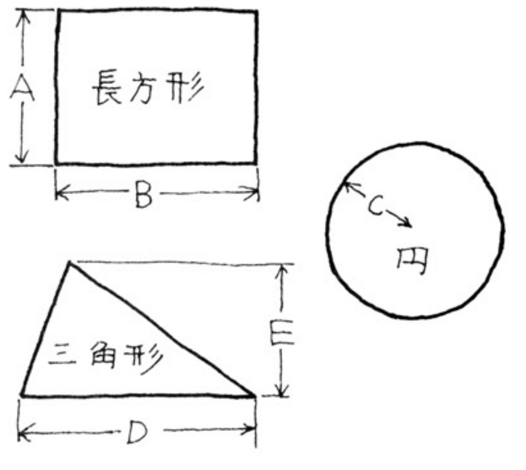
INPUT文の場合と同様に、変数の並び方と数値の並び方は一致していなくてはなりません。

```
READ  A, B, C, D
DATA  35, 6, -8, 7+5
```



READ、DATA命令を使って長方形、円の面積、三角形の面積を求めるプログラムはいがいと簡単につくれます。

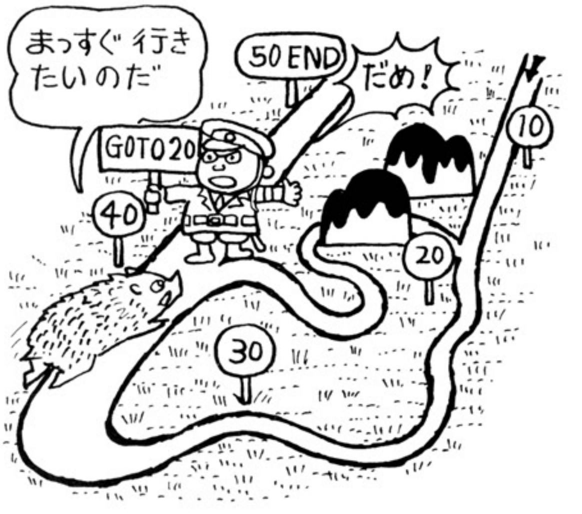
```
10  READ  A, B
20  S1=A*B
30  PRINT"チョウホウケイ=";S1
40  READ  C
50  S2=π*C↑2
60  PRINT"エン=";S2
70  READ  D, E
80  S3=D*E/2
90  PRINT"サンカクケイ=";S3
100  DATA  2, 4, 6, 8, 10
110  END
```



このプログラムについては、まだまだいろいろな工夫の余地がありそうですね。あなたもいろいろと試してみてください。

GOTOにはさからうな

今までに出てきたプログラムでは、コンピュータは、文番号の小さい方から大きい方へと順序正しく忠実に実行していきました。
しかし、ときにより順序を変えて実行させたいときが出てきます。こんなときにGOTO命令は強力な働きをしてくれます。




```
10  N=1
20  PRINT  N
30  N=N+1
40  GOTO  20
50  END
RUN
  1
  2
  3
  ⋮
```

GOTOは、そのあとに続く数値の文番号の命令文の所に、無条件に進めという意味です。

止まらない?…… SHIFT キーを押して、BREAK キーをたたいてやれば止まるはずです。

そこで問題をひとつ。
曽呂利新左衛門が大変な手柄をたてたとき、太閤秀吉は喜んで、
　秀吉「あっぱれなはたらきにつき、ほうびをつかわすぞ。望みをいってみよ」
　新左「かたじけないお言葉、恐れいりまする。さざいながら今日は1文、明日には2文、明後日には4文と毎日倍のごほうびを30日間だけ頂きとうございます」
なんとちっぽけな望みだわい、と軽く許した秀吉はいったいどれ位の金をほうびにくれたらいいのでしょうか。

```
10  D=1:F=1:S=1
20  PRINT"ニッスウ","ソノ日ノホウヒ゛","ホウヒ゛コ゛ウケイ"
30  PRINT  D;"日",F,S
40  D=D+1…………1日ごとに1だけ増やすため
50  F=2*F………………ほうびを倍々にするため
60  S=S+F……………いままでの合計を出すため
70  GOTO  30 ⇐ 注目
RUN
ニッスウ      ソノ日ノホウヒ゛         ホウヒ゛コ゛ウケイ
  1日        1                     1
  2日        2                     3
  ⋮          ⋮                     ⋮
  10日       512                   1023
  ⋮          ⋮                     ⋮
  20日       524288                1048575
  ⋮          ⋮                     ⋮
  30日       .53687091E+09         .10737418E+10
```

72〜73ページを見てください。



100文が1両なので10日目で1023文、20日目で10485両75文、30日目にはおどろくべし、およそ1000万両のほうびを要求していたわけになるのです!

ＩＦ……ＴＨＥＮの世界

ますます、コンピュータをコンピュータらしく使う命令の登場です。




```
10  IF△△△THEN□□□
20  ■■■
```

もし△△△の条件が満たされるのであれば、□□□の仕事を実行すること。もし満たされなければ、□□□は無視して、つぎの文番号の■■■に進め、というのがＩＦ～ＴＨＥＮ～の命令です。
□□□が数値の場合には、その数値の文番号にジャンプすることになります。

```
10  READ  A
20  IF  A>=0  THEN  PRINT"A=";A
30  GOTO  10
40  DATA  -10, 20, 5, -9, 8, -6, 5
50  END
RUN
A=  20
A=  5   <== 正の数だけ表示されている
A=  8
A=  5
```

ＩＦ……ＴＨＥＮ……の命令文の一般形は

ＩＦ　条件　ＴＨＥＮ　命令文または文番号

という形をとります。そして、ここでいう条件とは、等号や不等号であらわされた大小比較の式をいいます。

条件の記号	使い方
＝	A＝B
＜	A＜B＋C
＞	A＞B＋C
＜＝ または＝＜	A＋B＜＝C
＞＝ または＝＞	A＞＝B
＜＞ または＞＜	A＜＞B



IF……THENとその仲間たち

……の条件が真（YES）なら、THEN以降の命令が実行され、偽（NO）なら次の文番号へ進みました。ひとつこんどはほかの仲間たちも紹介してみましょう。

IF……THEN　GOTOまたはIF……GOTO
IF……THEN　PRINTまたはIF……THEN　?
IF……THEN　A=5*7のような代入文
IF……THEN　INPUT
IF……THEN　READ
IF……THEN　GOSUB
IF……THEN　RETURN
IF……THEN　STOP
IF……THEN　END




```
10 PRINT"🅲"
20 PRINT"1 カラ 9 マテ゛ ノ ニンイ ノ カス゛ ヲ イレテクタ゛サイ。
   チュウシ スル トキ ハ 0 ヲ イレテ クタ゛サイ"
30 L=0: M=0: N=0
40 INPUT A
50 IF A=0 THEN 90
60 IF A<=3 THEN L=L+1:GOTO 40
70 IF A<=6 THEN M=M+1:GOTO 40
80 N=N+1: GOTO 40
90 PRINT"アナタ ハ 1カラ3マテ゛ ノ カス゛ ヲ ";L;"カイ"
100 PRINT"ソシテ 4カラ6マテ゛ ヲ ";M;"カイ"
110 PRINT"7カラ9マテ゛ ヲ ";N;"カイ イレマシタネ"
120 END
```

こんなプログラムはいかがでしょうか？THENのあとには、いろいろな命令文をつけることができるので、工夫次第でちょっとした楽しいゲームのプログラムを作ることも可能です。

文番号10には、また新しい記号🅲が使われています。
クォーテーションマーク"　"の中で
SHIFTキーを押しながらCLR HOMEキーをたたくと、🅲を表示することができます。これはテレビ画面上の文字をすべて消してから、カーソルを画面の左上隅に移す機能を持っております。
またクォーテーションマーク"　"の中でただ単にCLR HOMEキーをたたくと🅷の記号が出ます。これはカーソルだけを左上隅に移すためのものです。
もしはっきりしないなら、PRINT"🅲"とか、PRINT"🅷"で確かめてみるとわかりますよ。



判断するならＩＦ……にまかせ

出てこい！　偶数……

ＩＦ……ＧＯＴＯを使って、たくさんの数の中から偶数だけを選びだすプログラムを考えてみました。ＩＦ……の数を見分ける力はこころにくいものです。

```
10 READ X:IF X=-9999 THEN STOP
20 IF X/2<>INT(X/2) GOTO 10
30 PRINT X;:GOTO 10
40 DATA 2,13,56,55,4,78,31
50 DATA 6,22,15,19,80,11,-9999
RUN
 2 56 4 78 6 22 80
```

文番号２０にあるＩＮＴ（Ｘ／２）はＸ÷２の整数部分のみを取りだす命令でしたね。したがって、もしＸが偶数ならＸ／２＜＞ＩＮＴ（Ｘ／２）は成立せず、文番号３０に進むわけです。また成立したら、それは奇数だと判断して、次の数値を読みとりにゆくのです。
では腕だめしに応用を考えてみましょう。
３の倍数とか４の倍数とかはどうすれば判断できるのかな？
もうおわかりですね。答はこうです。

３の倍数のための変更……

```
20 IF X/3<>INT(X/3) GOTO 10
```

４の倍数のための変更……

```
20 IF X/4<>INT(X/4) GOTO 10
```

出てこい！　最大値と最小値……

```
10 S=999:L=-999
20 READ X:IF X=-9999 THEN 80
30 IF X>L THEN L=X
40 IF X<S THEN S=X
50 GOTO 20
60 DATA 2,-5,91,256,-43
70 DATA 87,321,-76,-9999
80 PRINT"ｻｲﾀﾞｲﾁ=";L
90 PRINT"ｻｲｼｮｳﾁ=";S
100 END
RUN
ｻｲﾀﾞｲﾁ= 321
ｻｲｼｮｳﾁ=-76
```



文番号１０が大事な考え方です。最小値を入れるための変数Ｓにはできるだけ大きな数をはじめに入れておくのです。そして最大値のための変数Ｌにはできるだけ小さな数を！ところが実行結果はどうでしょう。変数ＬとＳは、ほんとうの最大値と最小値に化けて出てくるんですよ。これはＩＦ……ＴＨＥＮのうまい使い方といえます。

数の合言葉を捜しだせ……

最大公約数とはふたつの整数の合言葉です。たとえばふたつの数が10と20とすれば、どちらも割り切ることができるのは1，2，5，10の4個の数ですね。このうちで最大の数、つまり10が10と20の最大公約数と呼ばれるのです。
では、さっそくプログラムを作りあげましょう。

```
10  PRINT"X", "Y", "アイコトバ"
20  READ X,Y
30  PRINT  X,Y,
40  Q=INT(X/Y)
50  R=X-Q*Y
60  X=Y:Y=R
70  IF  R>0  THEN  40
80  PRINT  X:GOTO  20
90  DATA  63,99,1221,121
100  DATA  123456789,987654321
110  END
RUN
```

Yのうしろの","は続けてテレビ画面に表示するのにすごく便利です。



プログラムのタネあかし！
実はずっと昔、ギリシャのユークリッドという人が考えついた解き方です。

ふたつの数をXとYとする
↓
XをYで割って商Qと余りRを求めると
X=Q*Y+R
↓
余りはゼロか　　IF……
- はい → 答えはそのときのY
- いいえ → つぎに新しいXとしてY，新しいYとしてRの数を決める。（→ 「XをYで割って…」に戻る）

IF……を使って　どんどん　試してみること

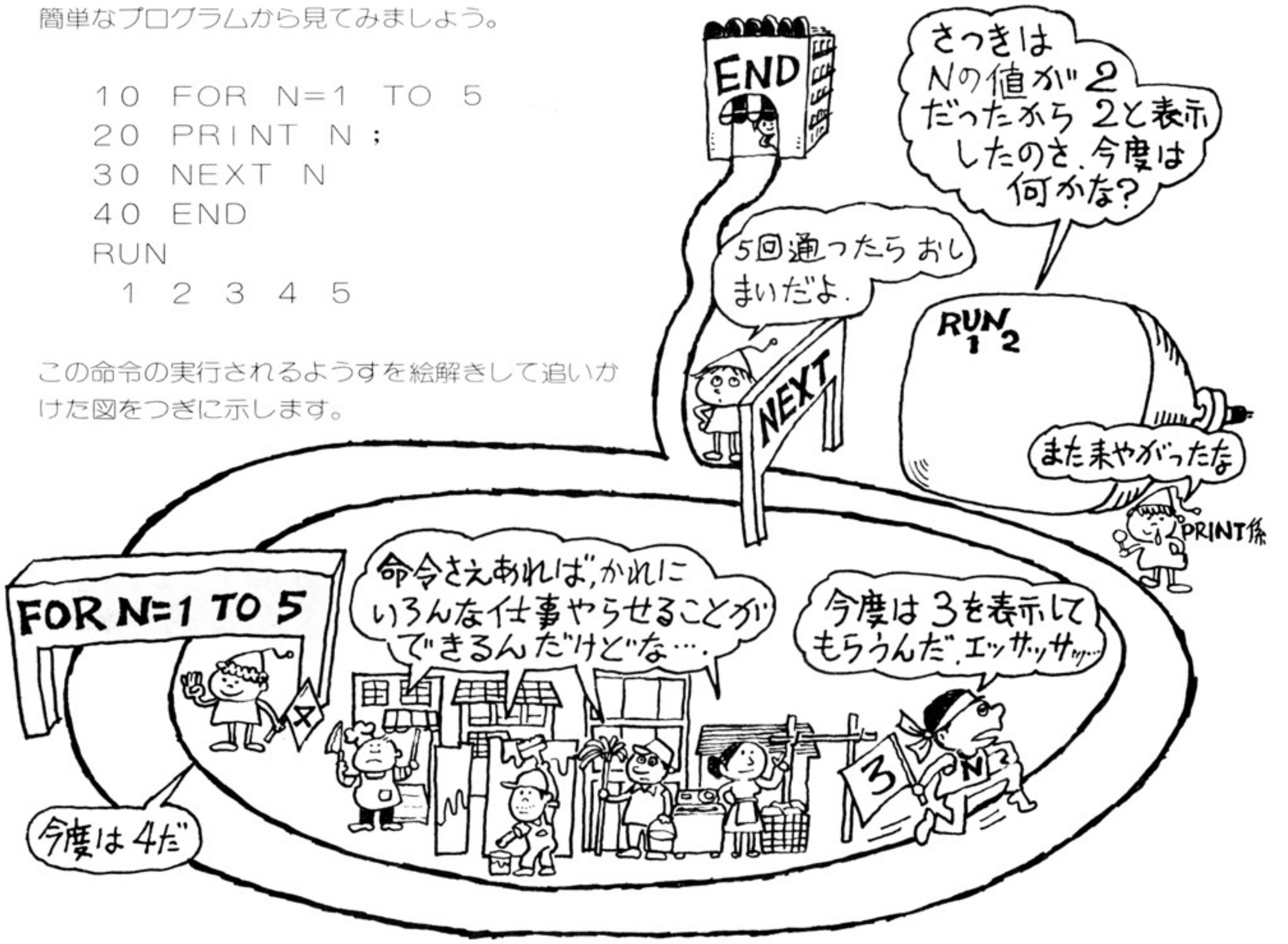
```
10  IF  SGN(X)=-1  THEN  ?"マイナス"
20  IF  SGN(X)=0  THEN  ?"ゼロ"
30  IF  SGN(X)=1  THEN  ?"プラス"
```

FOR...NEXTは繰り返しの達人

FOR……NEXT命令は、一定のルールのもとで繰り返し処理を進めるときに使う命令で、コンピュータを自由に扱うためのもっとも重要な命令のひとつです。
簡単なプログラムから見てみましょう。

```
10 FOR N=1 TO 5
20 PRINT N;
30 NEXT N
40 END
RUN
 1 2 3 4 5
```

この命令の実行されるようすを絵解きして追いかけた図をつぎに示します。



Nの変化を1、2、3と1ずつ大きくするのではなく、たとえば0.5ずつ大きくするとか、あるいは、2ずつ小さくすることもできます。このときの変化幅をSTEPという語で指定します。

0.5ずつ大きくするには

```
10 FOR N=1 TO 5 STEP 0.5
```

2ずつ小さくするには、たとえば

```
10 FOR N=5 TO 1 STEP -2
```

とすればいいのです。

FOR……NEXT命令の一般の形をつぎに示します。

FOR 変数＝初期値 TO 最終値 STEP 変化幅 繰り返されるプログラム NEXT 変数	⇦	初期値、最終値、変化幅は変数でも、定数、数式でもかまいません。

ループの中にループが見える

エリスはいま宿題をやっています。九九の計算表を作るのですが、コンピュータでやってしまおうと、FOR……NEXTを二重に使ってみました。

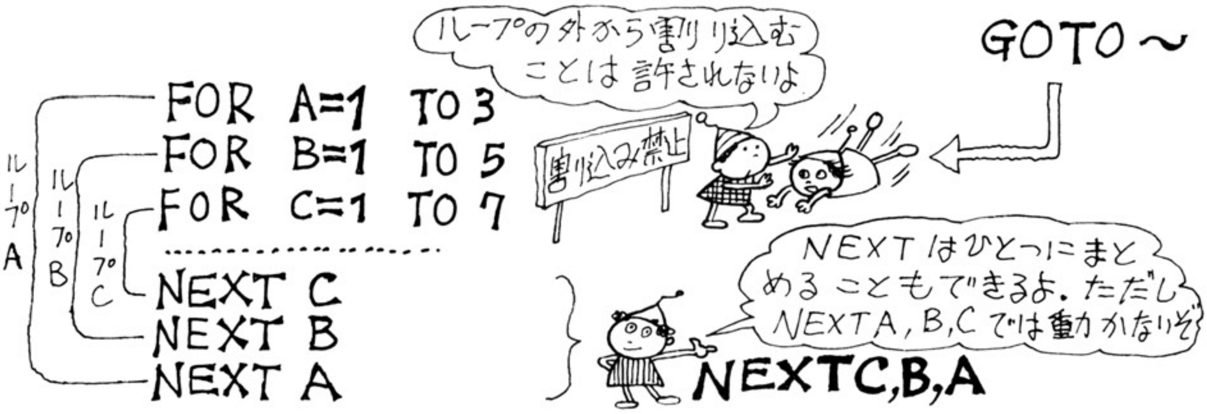



```
10 FOR X=1 TO 9
20 FOR Y=1 TO 9
30 PRINT X*Y;
40 NEXT Y
50 PRINT
60 NEXT X
```

変数Yのループ
変数Xのループ

変数XのFOR……NEXTループの中に、変数YのFOR……NEXTループがはさまれています。変数XとYはそれぞれ1から9まで変化しますが、最初に変数Xには1が代入されて、変数Yのループを実行します。つまり変数Xが1のまま変数Yが1，2，3……と9になるまで変化していき、そのつど文番号30で変数Xとの掛け算の結果が表示されていくのです。変数Yが9までいくと文番号50で行替えをして文番号60で今度は変数Xが2となるのです。

FOR……NEXTループは2重3重と重ねて使うことができます。しかし必ず守らなければならないことは、ループ同志を交差させないということ、そしてGOTOなどによって外からループ内に飛び込まない、ということです。



このようにループCは完全にループB内にあり、ループBは完全にループA内に含まれるようにします。右のようにNEXTを一語だけ使うてもあります。

♥

エリスがコンピュータで勉強していたら、こんな一通の手紙が届きました。
いったいどんな手紙なのでしょう。

```
KISS ME!                   〈1〉
10 ?"CELICE!ELICE!"
20 INPUT"ﾄﾞﾚﾀﾞｹｽｷ?";L
30 INPUT"ﾄﾞﾚﾀﾞｹｱｲﾀｲ?";V
40 FOR I=1 TO L
50 ?"I LOVE YOU"
60 FOR J=1 TO V
```

```
                           〈2〉
70 ?"CHU! ";
80 NEXT J:?:?
90 NEXT I
100 ?"YOUR SINCER
    ELY PRINCE"
110 END
```

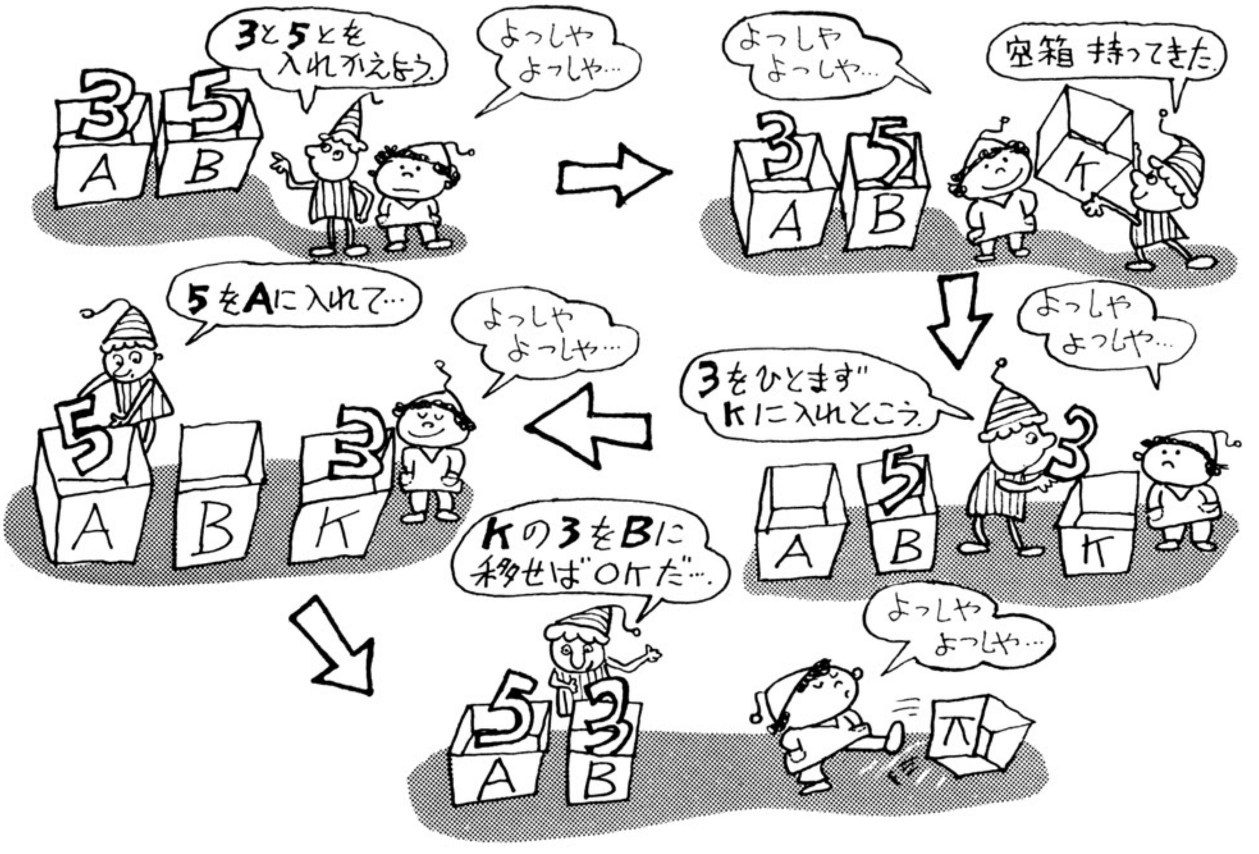


大きい順に並んでおくれ

任意に選んだ4個の数字を入れてやると、コンピュータは、その数字を大きい順に並べ替えてくれるプログラムです。INPUT命令を使ってみましょう。

```
10 PRINT"■"
20 PRINT"4コ ノ カス゛ ヲ オシエテ クタ゛サイ":PRINT
30 INPUT A,B,C,D
40 IF A<=B THEN K=A:A=B:B=K
50 IF B<=C THEN K=B:B=C:C=K
60 IF C<=D THEN K=C:C=D:D=K
70 IF A<B GOTO 40
80 IF B<C GOTO 40
90 IF A<C GOTO 40
100 PRINT A,B,C,D
110 PRINT:PRINT"モウイチト゛ ト゛ウソ゛":PRINT
120 GOTO 30
```

文番号40の命令文に注意してください。THEN命令以後で、別の変数Kの箱を使って、AとBとの数値を入れ替えてしまう仕事をやっています。もしはじめにA=3、B=5だったとしたら、



この仕事により、A=5、B=3になってしまいましたね。同様な処理を文番号50、60でも実行しています。文番号70~90は、この入れ替え作業を何回も繰り返させるために用意した命令文です。

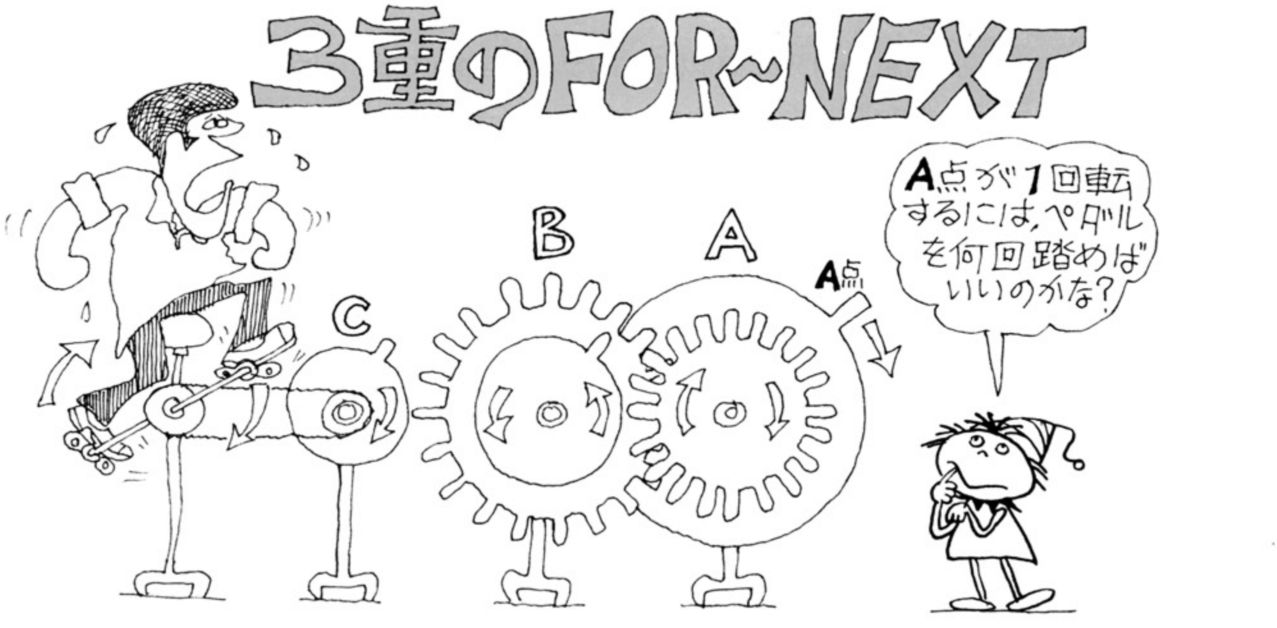
直角三角形はなんぼできる

三平方の定理$A^2=B^2+C^2$を満足する1から20までの正の整数をさがし出すプログラムを作ってみましょう。

```
10  PRINT"■"
20  PRINT"        |\"
30  PRINT"        | \"
40  PRINT"     B  |  \A"
50  PRINT"        |   \"
60  PRINT"        |____\"
70  PRINT"        ------"
80  PRINT"          C"
90  PRINT:PRINT"サンヘイホウ ノ テイリ ヲ ミタス セイ ノ セイスウ"
110  PRINT:PRINT"➡A","➡B","➡C"
120  FOR A=1 TO 20
130  FOR B=1 TO 20
140  FOR C=1 TO 20
150  IF A*A-B*B-C*C=0 THEN PRINT A,B,C
160  NEXT C,B,A
180  END
```



文番号10の意味は、もうご存知ですね、文番号20～80の間で、きれいな三角形が完成するように、うまく作図してみてください。
文番号120～160で、FOR～NEXTのループが三重になっています。A=1、B=1のときCを1から20まで、つぎにA=1、B=2のときCを1から20まで……のように、合計20×20×20=8000回も、文番号150に示す計算式を繰り返すわけです。したがって、この演算処理がひととおり完成するには、かなりの時間がかかるはずです。



PRINT文をたすける三銃士

ＰＲＩＮＴ文で、データを表示させる場合のカーソル操作などを行う３つの命令を説明します。



ＣＵＲＳＯＲコマンドは、テレビ画面の任意の位置へカーソルを移動させるもので、テレビ画面上を、Ｘ軸（水平）方向に、左から右へ０～３９、Ｙ軸（垂直）方向に、上から下へ０～２４にそれぞれ番号づけをしており、その値を数値または変数で指定します。

```
        ┌水平軸  ┌垂直軸
CURSOR   X   ,   Y
```

表の中の要素として、あるデータをその位置に表示させるような場合、ＰＲＩＮＴ文の前にこのＣＵＲＳＯＲコマンドを置いて表示位置を定めておくとよいわけです。

ＴＡＢ（　）および、ＳＰＣ（　）は、いずれもＰＲＩＮＴ文の中で使用するもので、ＴＡＢ（　）は、カーソルを、現在あるラインの左端から数えて、どれだけのタブレーションをとるかを指示するものです。たとえば、

```
PRINT TAB(8);"ABC"
```

というコマンドを与えると、ストリング"ＡＢＣ"は、左端から数えて（　）の中の数、つまり８文字分置いた後の９文字目から表示されることになります。



ＴＡＢ（　）の、（　）の中には、０から７８までのタブレーションが指定できます。また、その値をＣＵＲＯＲコマンドと同様に、変数で指定することもできます。

ＦＯＲ～ＮＥＸＴ命令と組み合わせて簡単なプログラムを走らせてみましょう。

```
10 FOR X=1 TO 20          10 FOR Y=1 TO 20
20 PRINT TAB(X);"*"       20 PRINT TAB(20-Y);"*"
30 NEXT X                 30 NEXT Y
```

ＳＰＣ（　）は、（　）内で示した数のスペースを意味します。つまり、ＳＰＣ（８）＝"←スペース８コ→"の関係があります。（　）内の値は、０から２５５までの数値あるいは、その間の値をとる変数でなくてはなりません。

ＴＡＢ（　）とＳＰＣ（　）を使った次の例で、両者のちがいを調べてみて下さい。

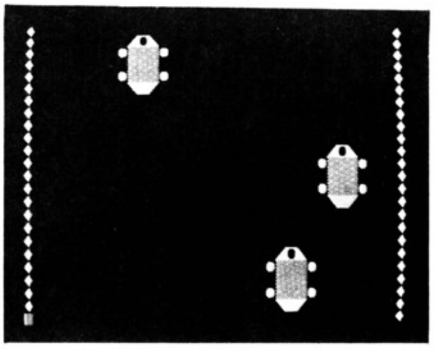
```
10 PRINT "🅲";
20 FOR X=1 TO 20 :PRINT TAB(8);
30 FOR Y=1 TO 20 :PRINT "*";
40 NEXT Y :PRINT :NEXT X
50 PRINT "🅷":FOR Z=1 TO 18
60 PRINT TAB(9);SPC(Z)
70 NEXT Z
```

RESTOREでピット・イン

自動車レースに挑戦……

もっともシンプルな自動車レースのプログラムはいかがですか？

```
10  X=33*RND(1)     ←── 乱数を作り出す命令です
                         (74ページ)
20  FOR A=1 TO 5
30  READ M$
40  PRINT TAB(0);"◆";TAB(X);M$;
50  PRINT TAB(37);"◆"
60  NEXT A
70  Y=10*RND(1)
80  FOR A=1 TO Y
90  PRINT TAB(0);"◆";
100  PRINT TAB(37);"◆":NEXT
110  RESTORE:GOTO 10
120  DATA" ◢●◣","●▒▒▒●"
130  DATA" ▒▒▒","●▒▒▒●"
140  DATA" ◥■◤"
```



文番号40のTAB（X）で道路の左側からどれだけのところに自動車を表示するかを決めています。エッ、車間距離は？……それは文番号70で0～9までの乱数を作り出し、文番号80から100で衝突しないようにしているはず。
ところで、文番号110のRESTOREは見なれない命令ですね。

RESTOREはDATAの先頭に戻すもの

DATA命令はプログラムのどこにあっても、またどんなに散在していても、READ命令に読みこまれていきます。

OK

```
10  DATA 27
20  READ A, B, C
30  DATA 10
40  ……
50  DATA 9, 13
60  READ D
70  END
```

ダメ

```
10  READ A, B
20  READ C
30  DATA 27, 10, 9
40  READ D
50  END
```



なぜダメかって？　変数Dに読みこむべきDATAの数値がないからですね。
そこでこうしたらどうでしょうか！

```
10  READ A, B
20  READ C
30  DATA 27, 10, 9
35  RESTORE
40  READ D
50  END
```

RESTORE命令は、プログラムの最初のDATA命令の、最初のデータから読みこめるようにしてくれます。

おしゃべりなストリング

コンピュータが人間にわかる言葉を作り出して、語りかけてくれたら……。そんな望みを満たしてくれるためにも、ぜひ必要な機能がストリング変数です。

```
10  A$="アナタ":B$="ワタシ"
20  C$=" ハ ":D$=" ヲ "
30  E$=" ニ ":F$="ガ "
40  G$=" デス"
50  I$="キライ":J$="スキ"
60  K$="アゲル":L$="ホン"
```



普通の変数記号のあとに、$（ドル記号）をつけたものをストリング変数といい、普通の変数の場合に大変よく似た処理が可能になります。以下いくつかの例を見ながら、その性質を調べてみましょう。

```
70  PRINT  B$
80  PRINT  A$
RUN
ワタシ
アナタ
```

「;」を使っていくつかのストリング変数をつなげてみましょう。

```
100  PRINT  B$;C$;A$;E$;L$;D$;K$
RUN
ワタシ ハ アナタ ニ ホン ヲ アゲル
```

もし「;」の代りに「,」を使ったらどうなるでしょうか？

```
120  PRINT  B$,A$,I$
RUN
ワタシ              アナタ              キライ
|←―――10字―――→|←―――10字―――→|
```

またいくつかのストリングを合成して新しいストリングにするには、ストリング変数どうしの足し算「+」を利用すればできます。ひとつためしてみましょう。

```
140  X$=B$+C$+A$+D$+J$+G$
150  Y$=A$+C$+B$+F$+I$+G$
160  PRINT  X$
170  PRINT  Y$
```



として、新しい変数を作り出してやることも可能です。

へたな俳句もかずうちゃ当たる



ストリング変数とINPUT命令とを組み合わせたプログラムで俳句を作ってみませんか?

```
10  INPUT  A$, B$, C$
20  PRINT  A$;" ";B$;" ";C$
30  GOTO  10
RUN
?フルイケヤ
?カワズ゛トヒ゛コム
?ミズ゛ノオト
フルイケヤ  カワズ゛トヒ゛コム  ミズ゛ノオト
```

五七五の各句を区切るためのスペース

INPUT記号を使ってストリングを入力するときには、クォーテーションマーク" "は必要ありません。

コンピュータを使って俳句をひとひねり。つぎのようなプログラムはいかがでしょう。

```
10  INPUT  AA$
20  PRINT  AA$;" ソレニツケテモ  カネノホシサヨ"
30  GOTO  10
```

RUNによって?と聞いてきたら、アキフカシでも何でも適当な5文字を入力してみてください。楽しい名句?がつぎつぎに表示されるはず。

ストリング変数は、READ、DATA命令と組み合わせたプログラムも可能です。

```
10  READ  X1$, X2$
20  PRINT  X1$;"スキ ト イウヒト ガ゛";X2$
30  DATA  アナタ ガ゛,  キライ デ゛ス
RUN
アナタ ガ゛スキ ト イウヒト ガ゛  キライ デ゛ス
```



READ命令を使った場合も、クォーテーションマークが必要ないことに注意してください。

一発豪快なプログラムをいかが?

```
10 WA$="ワッ":HA$="ハッ"
20 PRINT  WA$;
30 FOR  WH=1  TO  100:PRINT  HA$;
40 NEXT
50 END
```



LEFT$, MID$, RIGHT$

ＬＥＦＴ＄（　）、ＭＩＤ＄（　）、ＲＩＧＨＴ＄（　）は、いずれもストリングの中の一部を取り出して、新しいストリングを作るための命令です。

```
10  A$="ネ ウシ トラ ウ タツ ミ ウマ ヒツシ" サル トリ イヌ イ"
20  B$=LEFT$(A$,14)
30  PRINT  B$
RUN
ネ ウシ トラ ウ タツ ミ
```

左から１４番目までの文字

ＬＥＦＴ＄（Ａ＄，１４）は、Ａ＄にあたるストリングの中から、左から数えて１４番目までを拾い出して、新しいストリングを作り出す仕事をします。そしてここでは、その新しいストリングに、Ｂ＄というストリング変数が定義されたわけです。

ストリングの右から数えて、いくつかを拾い出すにはＲＩＧＨＴ＄（　）を使います。

```
40  C$=RIGHT$(A$,10)
50  PRINT  C$
RUN
サル トリ イヌ イ
```

右から数えて１０番目より終りまでの文字

ストリングの中間にある何文字かを拾い出すには、ＭＩＤ＄（　）を使うと便利です。

```
60  D$=MID$(A$,16,7)
70  PRINT  D$
RUN
ウマ ヒツシ"
```

左から１６番目の文字を起点に、その文字から７番目までの文字



LENはストリングの物さし

LEN（　）は、ストリングの文字数を知りたいときに使われる命令です。たとえば簡単な例では、

```
10 A$="ABCDEFG"
20 PRINT LEN(A$)
RUN
 7
```



となり、ストリング変数A＄の文字数つまり「7」が表示されるわけです。
LEN（　）を使って、正方形を描くプログラムを作ってみました。

```
10 PRINT "C":PRINT"* マークヲ ヨコニ ナラベテクダサイ"
20 INPUT A$
30 FOR I=1 TO LEN(A$)-2
40 PRINT TAB(2);"*";SPC(LEN(A$)-2);"*"
50 NEXT I
60 PRINT TAB(2);A$ :GOTO 20
```

INPUTする「＊」の数をいろいろ変えてみて下さい。コンピュータはLEN（　）によってその数を数えて作図を行うのです。
つぎに「＊」以外の文字・記号でも、作図できるようにしましょう。ここではLEFT＄（　）を使って前のプログラムの文番号20と40を変更します。

```
20 INPUT A$ : AA$=LEFT$(A$,1)
40 PRINT TAB(2);AA$;SPC(LEN(A$)-2);AA$
```

LENを使うとこんなストリングのパレードができます。

```
10 S$="SHARP BASIC"
20 FOR I=1 TO LEN(S$)
30 PRINT LEFT$(S$,I)
40 NEXT I
RUN
S
SH
SHA
SHAR
SHARP
SHARP
SHARP B
SHARP BA
SHARP BAS
SHARP BASI
SHARP BASIC
```

```
10 S$="SHARP BASIC"
20 FOR I=1 TO LEN(S$)
30 PRINT RIGHT$(S$,I)
40 NEXT I
RUN
C
IC
SIC
ASIC
BASIC
 BASIC
P BASIC
RP BASIC
ARP BASIC
HARP BASIC
SHARP BASIC
```

ASCとCHR$はしんせき同志

ASCについて…

```
10 PRINT ASC("A");
20 PRINT ASC("ABC");
30 T$="Z":PRINT ASC(T$)
40 END
RUN
 65 65 90
READY
```



ASCの（　）の中にストリングを入れてRRINTさせました。その結果はすべて数字になっています。実は、この数字はアスキーコードを示しているのです。コンピュータで用いられるすべてのキャラクタには、このアスキーコードがつけられています。124ページにその表がありますから参考にしてください。
ASC（　）は（　）内のストリングの最初の文字に対するアスキーコードを取り出してくれるものです。文番号10と20では（　）内のストリングが違っているのに、その結果が同じである理由もこのことでわかると思います。アスキーコードは、255以下の数です。

CHR$について…

キャラクターをアスキーコードに変換できるのなら、その逆の命令もあるのがあたりまえ。そのとおり、CHR$という命令がそれです。

```
PRINT CHR$(65), CHR$(ASC("K"))
A          K
READY
```



つぎに数字で暗号を作ってみました。CHR$に解読してもらいましょう。

```
10 FOR I=1 TO 24:READ A
20 B$=CHR$(A)
30 PRINT B$;:NEXT:END
40 DATA 73, 196, 83, 84, 85, 68
50 DATA 89, 196, 66, 65, 83, 73
60 DATA 67, 196, 79, 70, 196, 77
70 DATA 90, 45, 56, 48, 67, 46
RUN
I-STUDY-BASIC-OF-MZ-80C.
READY
```

STR$とVALは数字の変換

STR$について…




```
10  A=12:B=3:C=A+B
20  C$=STR$(A)+STR$(B)
30  PRINT  C, C$
40  END
RUN
 15           123
READY
```

STR$（A）によって、変数Aの値はそのまま数字列となりストリング処理されることになります。C$の内容が123となる理由もおわかりでしょう。つぎのプログラムではSTR$を使ってデータの「.」を合わせています。

```
10  FOR  X=1  TO  5
20  READ  A
30  L=5-LEN(STR$(INT(A)))
40  PRINT  TAB(L);A
50  NEXT:END
60  DATA  1.23456, 12.3456
70  DATA  123.456, 1234.56
80  DATA  12345.6
```

左のプログラムの結果はつぎのようになります。

```
RUN
     1.23456
    12.3456
   123.456
  1234.56
 12345.6
READY
```

VALについて…

VAL命令は、STR$命令のまったく逆の機能をもっています。つまり数字列のストリングを数値に変換するものです。




```
10  A$="123456"
20  B=VAL(A$)
30  C=654321+B
40  PRINT  A$
50  PRINT  B
60  PRINT  C
80  END
RUN
123456
  123456
  777777
READY
```

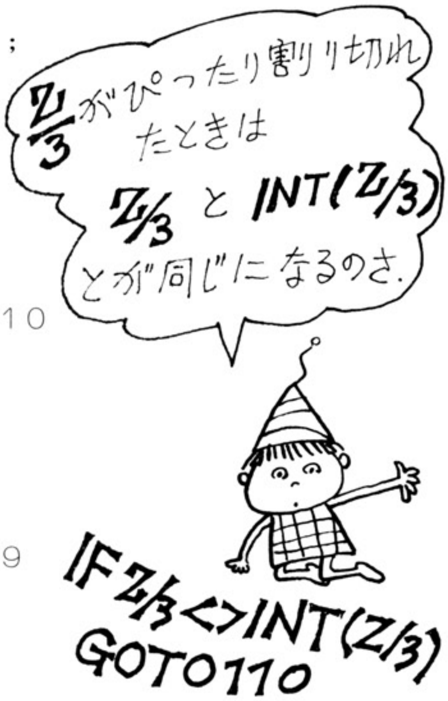
123456 ……………数値ではありません。ストリングです。

123456、777777 ……………数値なので数値の最上位の前に±の符号を入れるためにひとつスペースがあります。負のときは、そこにマイナス符号がはいります。

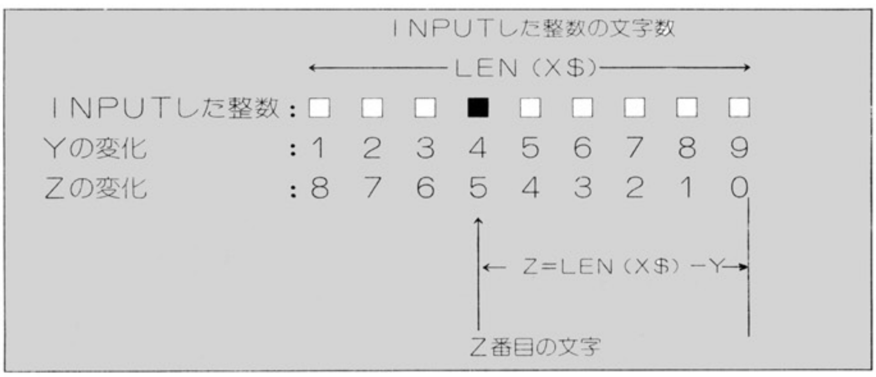
※願いまして¥123,456,789円也…

任意のけた数の整数をINPUT命令で読みとって、右から3けたごとにコンマ「,」をつけて書きだすプログラムです。整数として0を与えたら完了するはずです。

```
10 PRINT"セイスウ ヲ カイテ クタ゛サイ";
20 INPUT X$
30 IF X$="0" THEN STOP
40 PRINT"¥";
50 FOR Y=1 TO LEN(X$)
60 PRINT MID$(X$,Y,1);
70 Z=LEN(X$)-Y
80 IF Z/3<>INT(Z/3) GOTO 110
90 IF Z=0 GOTO 110
100 PRINT",";
110 NEXT Y
120 PRINT"円":GOTO 10
RUN
セイスウ ヲ カイテ クタ゛サイ? 123456789
¥123,456,789円
セイスウ ヲ カイテ クタ゛サイ? 1234
¥1,234円
```



文番号80で、右から数えてZ番目の文字が3の倍数のけたであるかどうかを確かめています。もし3の倍数であれば文番号100の命令文で「,」を入れることにしたわけです。たとえば入力した整数を9けたとしてこんな具合に。



右から読んでも、左から読んでも同じ言葉になるかどうかを確かめるプログラムを作ってみましょう。もちろん、文字でも数字でもかまいませんが、濁点、半濁点を使うと、おかしな所にこれらの記号がついてしまいます。

```
10 PRINT"C"
20 PRINT"ミキ゛カラ ヨンテ゛モ ヒタ゛リカラ ヨンテ゛モ・・・?"
30 INPUT X$:PRINT
40 FOR Y=LEN(X$) TO 1 STEP -1
50 PRINT TAB(2);
60 PRINT MID$(X$,Y,1);:NEXT Y:PRINT
70 GOTO 30
```

いま、何どきだ？

江戸時代は、不定時法といって、日の出から日の入りまでの時間と、日の入りから日の出までの時間とを、それぞれ別々に６等分して時間を区切っていました。つまり同じ１刻（約２時間）といっても、夏と冬とでは長さが違っていたのです。このようにして１日の時間を12に分け、真夜中から順に、ね、うし、とら……と、12支の順に名称をつけていました。

草木も眠る丑満つ時などの言葉はここから来ており、今日の時間でいうと午前２時頃に当たります。また、各１刻はさらに半分に分け、前半を初刻、後半を正刻と称していました。

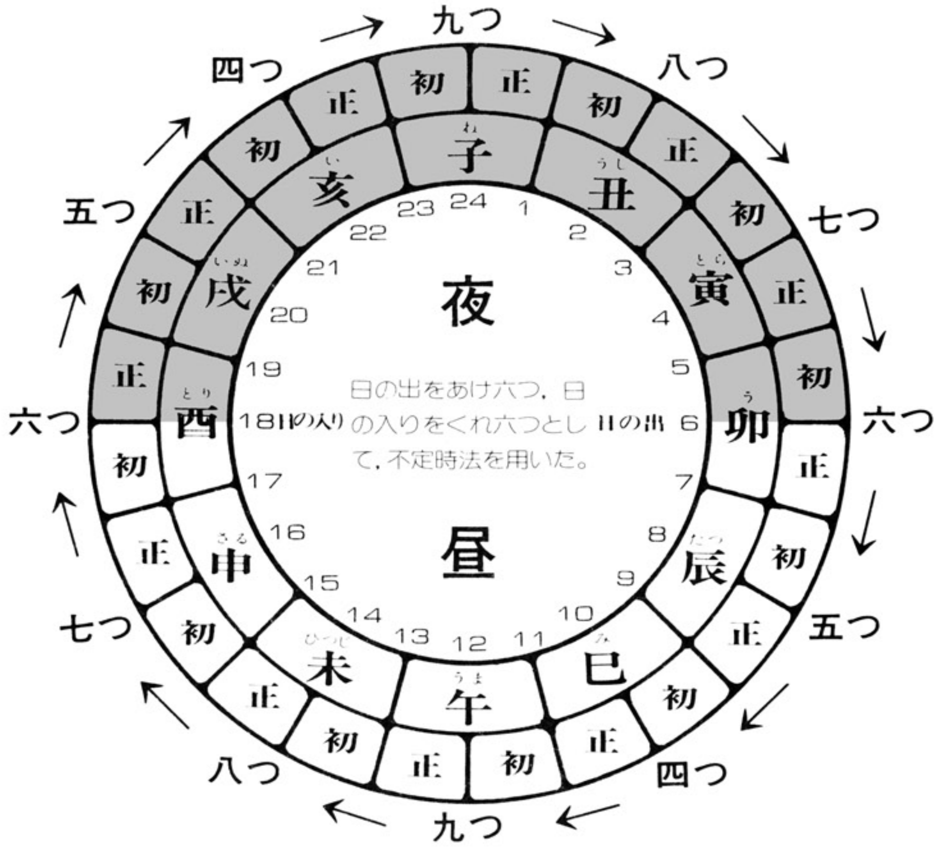
今日でも、お昼の１２時を正午というのは、その名残りです。

記録に残っている我が国最古の時計は水時計です。当時は、宮廷に漏刻博士という時間係りの役人がいて水時計で時刻を知り、たいこを叩いて、それを知らせていました。低い方に落ちる水の量で時間を計っていたのですから、当時の時刻は江戸時代より正確だったでしょうし、現代のように昼夜に関係なく１日を12等分する、いわゆる定時法によっていたわけです。

たいこの合図は、１刻ごとに９の倍数だけ叩くのが原則ですが、これでは回数が多すぎるので、たとえば9×3=27のとき、はじめの２を省略して１けた目の７だけの数を叩いていました。昔の時刻のあらわし方の、もうひとつの方法、九つとか八つなどが、時刻が進むにつれて数が減少していくのはこのためです。このやり方は、江戸時代にも引き継がれ、たいこの代りに鐘をついて時刻を知らせるようになりました。これを「時の鐘」といい、当時の人たちはこれによって時刻を知ったわけです。

なお、時の鐘は、今日でも東京近郊の川越市に昔のまま残っており、昔をしのぶことができます。

叩くべき
たいこの数
9×1＝ 9
9×2＝18
9×3＝27
9×4＝36
9×5＝45
9×6＝54
30分置きにも
たいこを叩いた
1／4刻…1
2／4刻…2
3／4刻…3



単利と複利でどうちがう?

銀行に１万円預けて１年後に１万600円だけ受けとったら、利率は600/10000＝0.06(6%)だとわかります。では２年間預けたら、利息はいくらついているでしょうか?
この場合は２通りの考え方があります。ひとつ目は１年間で600円の利息がついたのだから、２年間では２倍の1200円の利息になるという考え方です。これを単利の利息計算といいます。
ふたつ目は１年間で利息600円がついているから、２年目の初めには１万600円の預金になっている。そこで２年目の利息は10600×0.06＝636円になり、合わせて1236円の利息になるという考え方です。これを複利の利息計算といいます。
複利計算のほうが単利計算よりすこし多い利息になっていますが、もっと金額が大きかったり、もっと長い期間預けると、利息の差はもっと大きくひらくはずです。
それぞれの利息計算でｎ年目の元利合計を求める式はつぎのようになることがわかります。

単利による元利合計（ｎ年目、利率R）

$B = X(元金) + n \cdot X \cdot R$

複利による元利合計（ｎ年目、利率R）

$C = X \cdot (1+R)^n$



この式をもとに、単利と複利で元利合計を計算するプログラムをつくってみました。




```
10 PRINT"ガンキン ハ イクラ ";
20 INPUT X
30 PRINT"キンリ ハ ナン% ";
40 INPUT R
50 PRINT"ナン年 マデ ミマスカ ";
60 INPUT Y:PRINT:PRINT
70 PRINT"ガンキン=";X
80 PRINT"キンリ =";R;"%"
90 PRINT" 年";TAB(4);"タンリ";
100 PRINT TAB(15);"フクリ";
110 PRINT TAB(26);"ソノサ"
120 FOR A=1 TO Y
130 B=X+A*X*(R/100)
140 C=INT(10*X*(1+R/100)↑A)/10
150 D=C-B
160 PRINT A;TAB(4);B;
170 PRINT TAB(15);C;TAB(26);D
180 NEXT A
190 PRINT:PRINT:GOTO 10
```



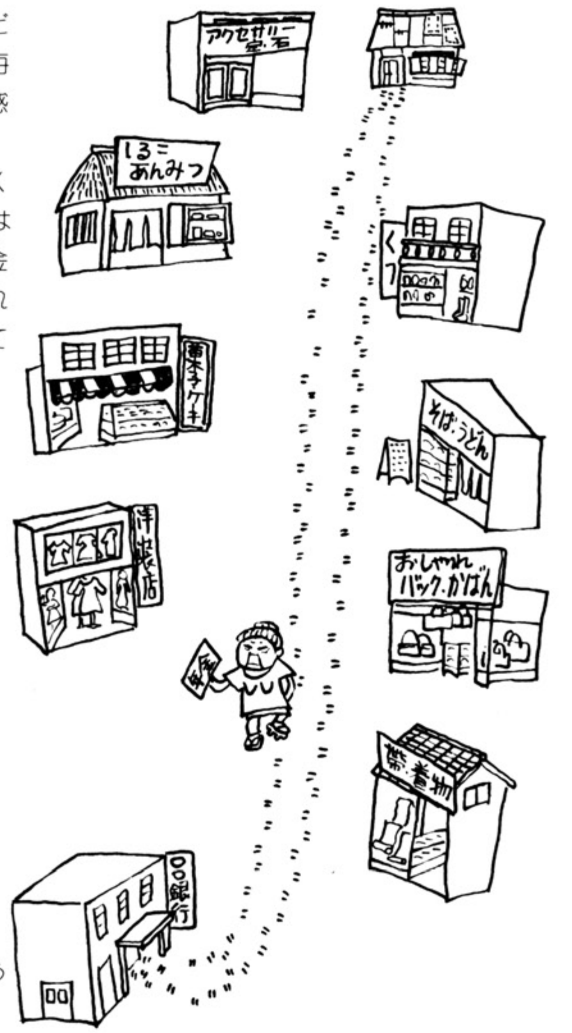
このプログラムを実行した具体的な例を示します。

ガンキン=10000
キンリ =6%

年	タンリ	フクリ	ソノサ
1	10600	10600	0
2	11200	11236	36
3	11800	11910.1	110.1
4	12400	12624.7	224.7
5	13000	13382.2	382.2
6	13600	14185.1	585.1
7	14200	15036.3	836.29999



年金を５年間預けておいたら…



前例では、一度お金を預けたとき、単利と複利で利息がどうちがうかをみたわけです。しかし、定期預金のように毎月毎月お金を預けるやり方が、私たちにはもっと身近に感じられます。

そこで、決まったお金Xを今年預けておけば、１年目はX（１＋R）、２年目はX（１＋R）^{2} という具合に元利合計は大きくなっていきます。それに加えて、毎年定期的にお金Xを預けると、１年後に預けたお金についても、預け入れから１年後、つまり今から２年後にはX（１＋R）となっているはず。この増え方のようすをつぎに示します。

１年後の元利合計（Xは元金、Rは利率）

$$M_1=X(1+R)$$

２年後の元利合計

$$M_2=X(1+R)^2+X(1+R)$$

３年後の元利合計

$$M_3=X(1+R)^3+X(1+R)^2+X(1+R)$$

こうやってｎ年後の元利合計を求めると

$$Mn=X(1+R)^n+X(1+R)^{n-1}\cdots+X(1+R)$$

となり、これをさらに簡単に表わすと

$$Mn=X((1+R)^{n+1}-(1+R))/R$$

さて、毎年同じ額を預けて、希望する年の元利合計がどうなっているかを示すプログラムを作ってみます。ただし、同じ額を預けるとしても、その最低額と最高額は入力できるるようにしておきました。

```
10 PRINT"キンリ ハ ナン% デスカ ";
20 INPUT R
30 PRINT"ガンキン ノ ハンイ ヲ ドウゾ"
40 PRINT"サイテイ ハ ";:INPUT L
50 PRINT"サイコウ ハ ";:INPUT H
60 PRINT"ナン年 マデミマスカ ";
70 INPUT Y:PRINT:PRINT
80 PRINT"キンリ=";R;" %"
90 PRINT"マイ年";TAB(12);Y;"年メ"
100 R=R/100
110 FOR A=L TO H STEP 10000
120 B=INT(A*((1+R)↑(Y+1)-(1+R))/R)
130 PRINT A;"円";TAB(12);B;"円"
140 NEXT A
150 PRINT:PRINT:GOTO 10
```

左のプログラムを実行した結果は

```
キンリ ハ ナン% デスカ ? 10
ガンキン ノ ハンイ ヲ ドウゾ
サイテイ ハ ? 50000
サイコウ ハ ? 100000
ナン年 マデ ミマスカ ? 6

キンリ= 10 %
マイネン        6年メ
 50000円        424358円
 60000円        509230円
 70000円        594101円
 80000円        678973円
 90000円        763845円
 100000円       848717円

キンリ ハ ナン% デスカ ? ■
```

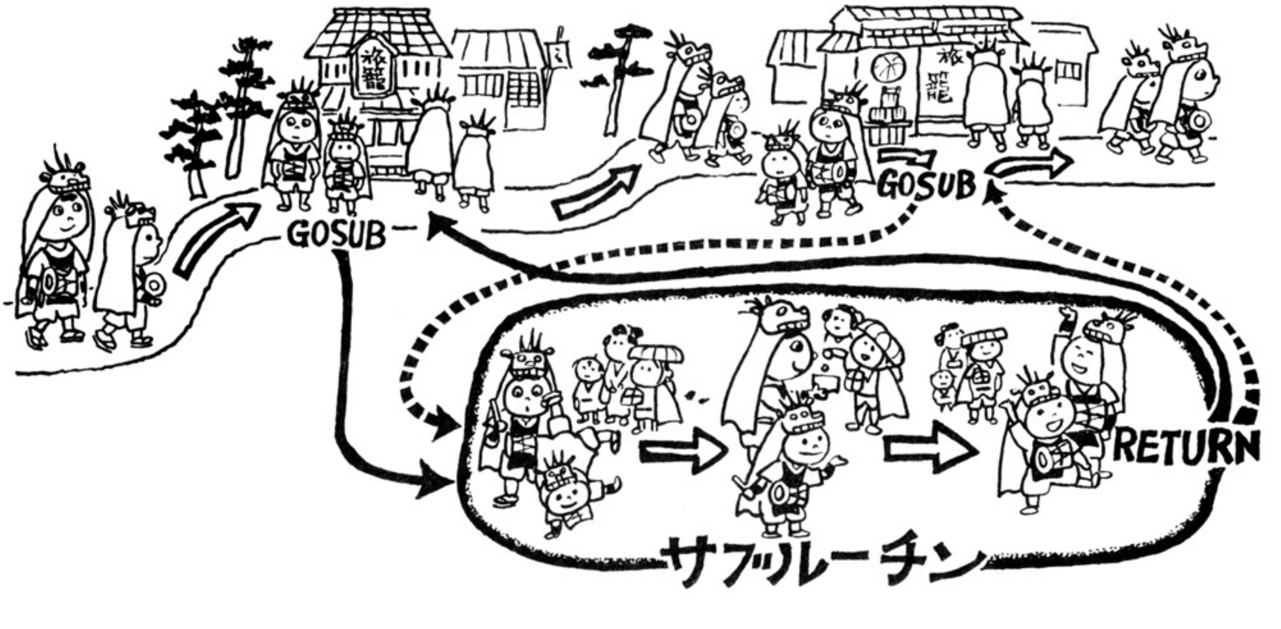
サブルーチンはプログラムの切札



プログラムの中で、同じ処理方法の仕事が何回も繰り返し出てくることがあります。このようなとき、その仕事だけを小プログラムとしてまとめておいて、主プログラムから必要に応じ何度でもそこに飛んで行って仕事をする方法があります。この小プログラムをサブルーチンといい、サブルーチンに飛んで行くための命令にGOSUBが使われます。GOSUBを使ったプログラムの例を次にあげておきました。

```
10  PRINT"Ⓒ"
20  PRINT  TAB(10);"**ウリアゲ ヒョウ**":PRINT
30  PRINT"       5     10     15     20     25     30   35"
40  PRINT"□□□□□□┏□□□□□┏□□□□□┏□□□□□┏□□□□□┏□□□□□┏□□□□□"
50  PRINT"□   オミソ":GOSUB 200
60  PRINT"□   オサトウ":GOSUB 200
70  PRINT"□   オサケ":GOSUB 200
80  PRINT"□   ショウユ":GOSUB 200
110  PRINT"□"
120  PRINT:END
200  PRINT"□";:READ A
210  FOR N=1 TO A:PRINT"▩";:NEXT N
220  PRINT A
230  RETURN
240  DATA 20,15,21,24
```

上記プログラムで、サブルーチンに当たる部分は、文番号200～230です。サブルーチンの終りににあるRETURN命令によって、処理はふたたび主プログラムに帰ります。



止めて、調べて、CONTする

プログラムを作ってRUNさせても、1回で思った通りに動いてくれるとは限りません。そんなとき、適当なところにSTOP命令をプログラムしておいて、止まったときの変数がどんな内容になっているか、確かめたいときがあります。たとえば、つぎのプログラムでは、

```
10 READ A,B
20 X=A*B
30 STOP
40 Y=A/B
50 END
60 PRINT X,Y
70 DATA 15,5
80 END
RUN
BREAK IN 30
```

CONT命令が使えるケース

- ■プログラムの実行をSHIFT － BREAKキーで中止したいとき
- ■STOP命令、END命令で実行を中止したとき
- ■INPUT命令でSHIFT － BREAKキーで入力を中止したとき

CONT命令が使えないケース

- ■RUNで実行する前のとき
- ■実行を止めたあと、プログラムを編集したとき
- ■実行中のエラーで"READY"に戻ってしまったとき
- ■カセットテープの操作をBREAKキーで中止したとき
- ■音楽を鳴らすMUSIC命令を中止したとき

となりますね。このときダイレクトモードで

```
PRINT A,B,X CR
```

と変数を表示させてプログラムのチェックができるはず。そこでプログラムをふたたび始めたいときは

```
CONT CR
```

とコンピュータに命令を与えてください。止まったところからコンピュータは実行を再開してくれます。そして、つぎの文番号50のEND命令で"READY"と表示し、また止まってしまいます。
こんどは、ダイレクトモードで

```
X=3:Y=5 CR
```

としてみたらいかがですか？…… CONT命令でまたプログラムを続行させたら、変数XとYは、3と5が表示されるでしょう。
つぎのプログラムは、メビウスの環のようにいつまでも＊マークの三角山を表示し続けるものです。

```
10 A=0:B=38:C=1
20 FOR X=A TO B STEP C
30 FOR Y=0 TO X
40 PRINT"*";
50 NEXT Y:PRINT:NEXT X
60 K=A:A=B:B=K
70 C=-C:GOTO 20
```



ここでSHIFTキーを押しながらBREAKキーをたたくと、実行を中止します。そこで今度は

```
100 END
```

とプログラムを追加してからCONT命令を与えます。すると

```
CONT ERROR
```

そうなんです。STOP命令、END命令あるいはBREAKキーで命令を中止したあとで文番号を使ったプログラムの編集をしたら、CONT命令は使えないことに注意してください。

ON…GOTO…でまとめて飛んで！

GOTO命令については、もうすでに何度も出てきました。ここでは、GOTO命令の機能を、もう少し拡大したON…GOTO命令について調べてみましょう。

ON 変数または数式 **GOTO** 文番号, 文番号, 文番号…………

この値が1のとき ―― 1番目の文番号
〃　2のとき ―― 2番目の文番号
〃　3のとき ―― 3番目の文番号

たとえば、ONの後の変数または数式の値が、3であればGOTO以下に並んでいる3番目の文番号に飛ぶことになります。つまり、変数の値にしたがってプログラムの分岐先を指定できるのです。

```
10  INPUT"ｶｽﾞﾊ?";A
20  ON A GOTO 50,60,70
50  PRINT"XXX":GOTO 10
60  PRINT"YYY":GOTO 10
70  PRINT"ZZZ":GOTO 10
RUN
ｶｽﾞﾊ ? 1
XXX
ｶｽﾞﾊ ? 2
YYY
ｶｽﾞﾊ ? ■
```

たとえば1.2と与えるとやはり1と小数点以下を切り捨てた整数として扱われます。



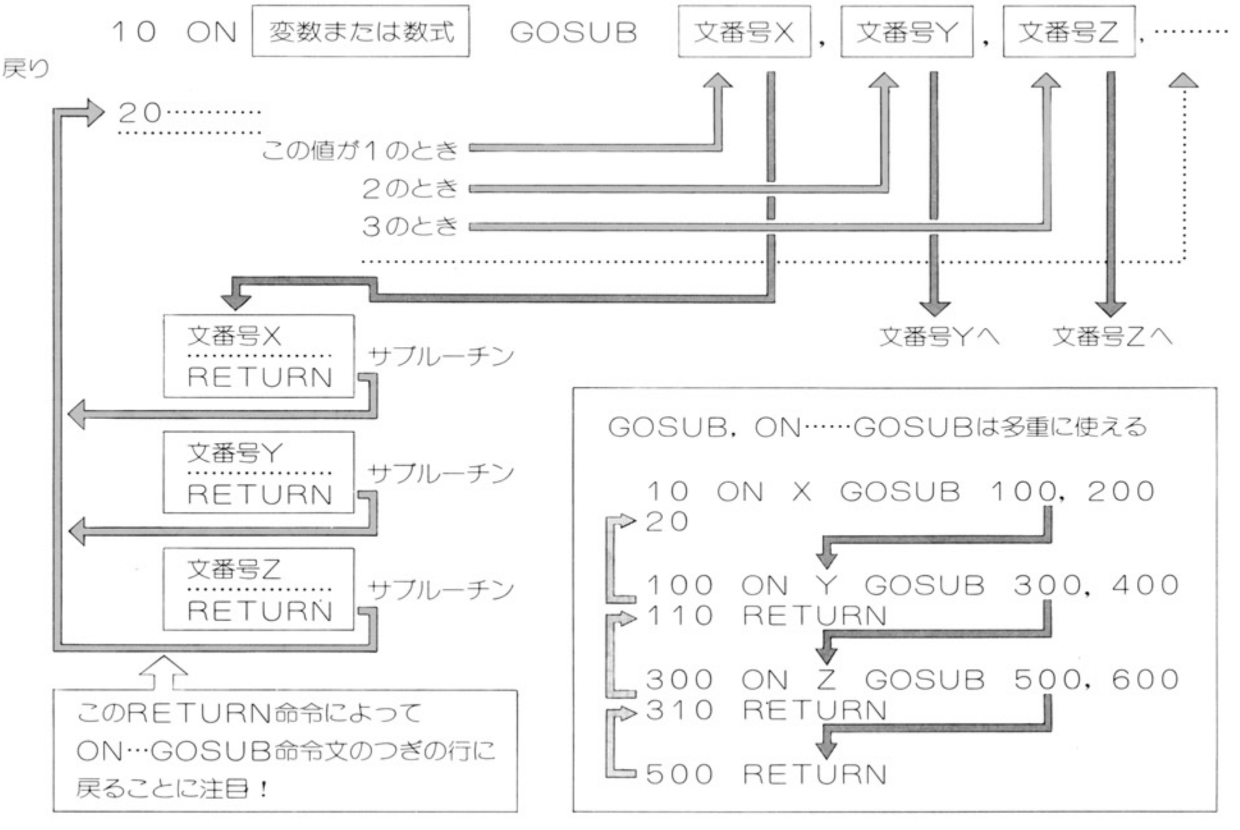
ここでON...GOTOを使ってババヌキゲームをやってみます。5枚の札の中に1枚だけ「ババ」が入っています。「ババ」はどの位置にあるかわかりません。そしてあなたは何枚めに「ババ」があるのか当ててコンピュータと勘比べするのです。
プログラム中で「パスをしますか」と聞いてきますからパスなら1、パスしないなら2、このゲームをやらないなら3をキー入力することにします。パスは3回までできます。

RNDは74ページ

```
10  R=INT(5*RND(1))+1
20  N=N+1:IF N=6 THEN 120
30  PRINT"ｺﾉｶｲ ﾊ ﾊﾟｽ ｼﾏｽｶ?":INPUT X
40  ON X GOTO 60,90,50
50  PRINT"ｺﾞｸﾛｳｻﾏ!!!":GOTO 120
60  NP=NP+1
70  IF NP>=4 THEN NP=NP-1:N=N-1:PRINT
    "ﾉｺﾘ ｾﾞﾝﾌﾞ ﾋｲﾃ ｸﾀﾞｻｲ"
80  GOTO 20
90  NR=NR+1
100  IF R=NR+NP THEN PRINT"ｼﾞｬｰﾝ!!  ﾊﾞﾊﾞ
    ｶﾞﾃﾞﾀ":GOTO 120
110  PRINT"ｾｰﾌ!!   ﾗｯｷｰﾃﾞｼﾀﾈ":GOTO 20
120  END
```

ON…GOSUBはサブルーチンの集団

ON…GOTOの働きとよく似た命令に、ON…GOSUB命令があります。



そこで腕だめしに時間割のプログラムを考えてみました。つぎのプログラムでとくに大切なことは、文番号90で文番号170～190のサブルーチンへ飛んでいるにもかかわらず、さらに文番号180でサブルーチンを呼び出している点です。
このようにGOSUBは、多重に使える便利な働き者なのです。

```
10 A$="ｺｸｺﾞ   ":B$="ｽｳｶﾞｸ  ":C$="ｴｲｺﾞ   "
20 D$="ﾘｶ     ":E$="ｵﾝｶﾞｸ  ":F$="ﾀｲｲｸ   "
30 G$="ｼｬｶｲ   ":H$="ﾋﾞｼﾞｭﾂ ":I$="ｷﾞｼﾞｭﾂ "
40 J$="ﾄﾞｳﾄｸ  ":K$="HR     "
50 INPUT"ﾅﾝﾖｳﾋﾞ?  ";X$      ← 日～土までを与えます。
60 X=ASC(X$)-111
70 FOR Y=0 TO 4:PRINT TAB(2+7*Y);Y+1;
80 NEXT Y:PRINT
90 ON X GOSUB 170,110,120,130,140,150,160
100 PRINT:GOTO 50
110 PRINT"月 ";A$;B$;D$;G$;K$:RETURN
120 PRINT"火 ";B$;E$;H$;H$;D$:RETURN
130 PRINT"水 ";C$;C$;I$;A$;F$:RETURN
140 PRINT"木 ";B$;D$;F$;G$;E$:RETURN
150 PRINT"金 ";A$;D$;I$;C$;C$:RETURN
160 PRINT"土 ";B$;G$;D$;K$:RETURN
170 FOR Y=1 TO 6
180 ON Y GOSUB 110,120,130,140,150,160
190 PRINT:NEXT Y
200 RETURN
```

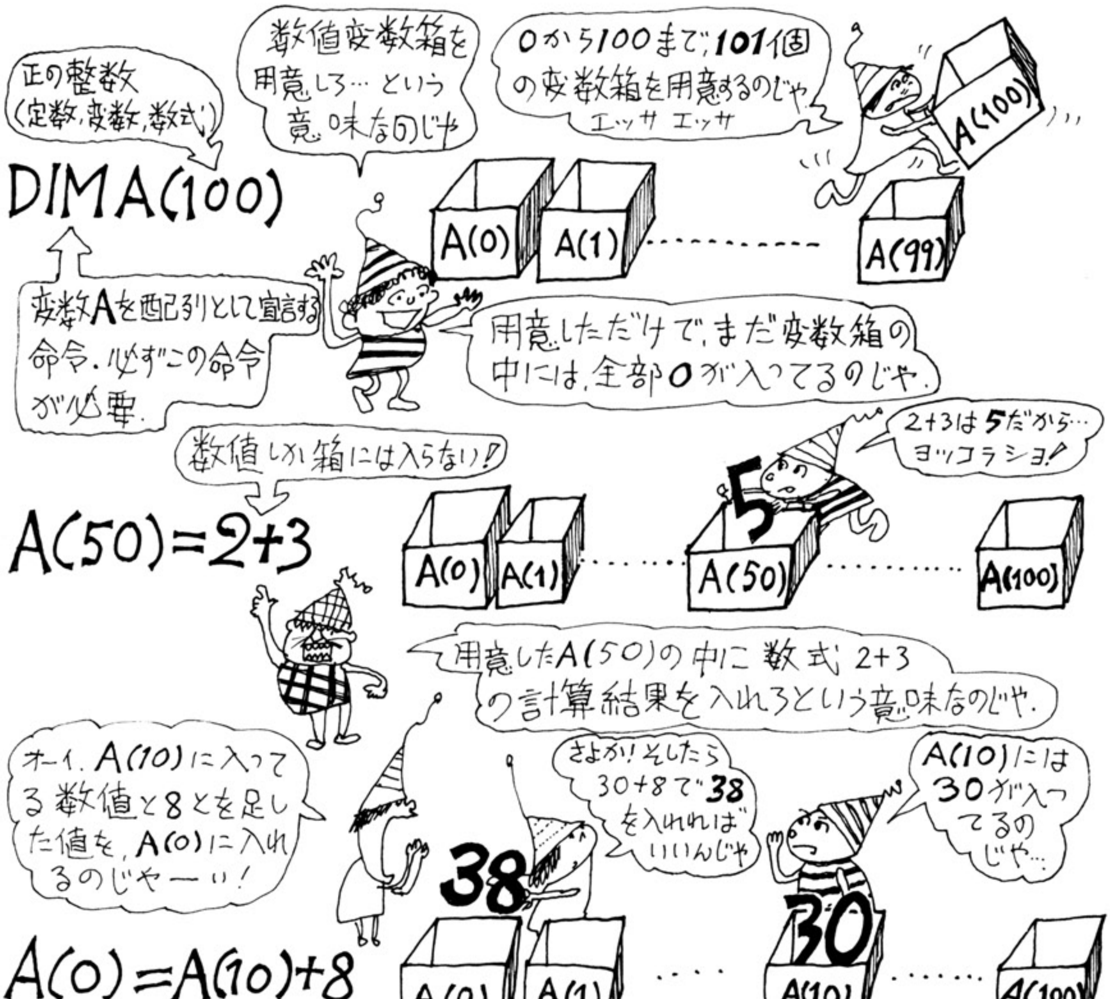


1次元配列は100人力

１００個のデータを変数に代入することを考えてみます。たとえば変数Ａ１とかＡ２とかを使うと……

```
10  A1=5
20  A2=30
30  A3=12
……………………
```

ちょっと待ってください。大変なことになりそうです。プログラム文を１００個も書かなくてはならないなんて！……こんなとき新しい変数のタイプとして１次元配列というものを用いると、とても便利になります。１次元配列とはどんなものかさっそくみてみましょう。



１次元配列というものがわかりましたか。配列を使って最初のプログラムをつくってみました。

```
10  DIM  A(100)
20  FOR  J=1  TO  100
30  READ  A(J)
40  NEXT  J
50  DATA  5, 30, 12, ………
```

どうです。非常に短いプログラムになりますね。この例でもわかるように配列の形をした変数は、たとえばＡ（Ｊ）というように添字つき変数のカッコのなかを変数Ｊで指定することができ、それが大きな武器となっているのです。

ストリング変数にも配列がある

数値変数に配列があるのなら、ストリング変数にも配列があってもよさそうなもの……
そうです。ストリング変数の場合も、同様に配列が使用できます。
ストリング変数の１次元配列とはどんなものか、ここで紹介します。



簡単なプログラムを作ってみました。
このように、いくつかのストリングを配列に入れておくと、そのストリングを使うたびに書き込む必要がありません。プログラム自体もすっきりしたものになります。

```
10 DIM A$(2), B$(2), C$(2)
20 FOR J=1 TO 2 :READ A$(J), B$(J)
30 C$(J)=A$(J)+" "+B$(J)
40 PRINT A$(J), B$(J), C$(J)
50 NEXT J
60 END
70 DATA YOUNG, GIRL, WHITE, ROSE
```

```
RUN
YOUNG         GIRL          YOUNG GIRL
WHITE         ROSE          WHITE ROSE
READY
```

配列はファイル作りの名人（？）

テストをするのはいいけれど、テストの結果を集計するのがめんどうだという先生。それならテストをやらなければよいという学生。そういう訳にもいかない先生のためによい方法を教えましょう。配列を使えば、もう悩むこともありません。
つぎに学生番号と数学の点数のデータがあります。

学生番号	20	15	12	40	23	16	31	45	26	11
点数	75	51	28	56	100	81	60	43	66	48

成績順に並びかえたファイルのプログラムを作ってみました。




```
10  DIM A(10), B(10)
20  FOR J=1 TO 10
30  READ A(J), B(J):NEXT
40  FOR K=1 TO 9:M=0
50  FOR J=K TO 10
60  IF B(J)<=M THEN 80
70  M=B(J):L=J
80  NEXT J
90  B(L)=B(K):B(K)=M
100  A1=A(L):A(L)=A(K):A(K)=A1
120  NEXT K
130  PRINT"C"
140  PRINT" セイセキ シ゛ュンイ （スウカ゛ク）"
150  PRINT
160  PRINT" カ゛クセイハ゛ンコ゛ウ";"      ";
170  PRINT"テンスウ"
180  FOR J=1 TO 10
190  PRINT A(J);TAB(14);B(J):NEXT J
200  END
210  DATA 20,75,15,51,12,28,40,56,23,100
220  DATA 16,81,31,60,45,43,26,66,11,48
RUN
 セイセキ シ゛ュンイ （スウカ゛ク）

 カ゛クセイハ゛ンコ゛ウ      テンスウ
 23                100
 16                81
 20                75
 26                66
 31                60
 40                56
 15                51
 11                48
 45                43
 12                28
READY
```



英語に挑戦する

よく単語帳をつかって英単語の勉強をしたものです。コンピュータを使ってもっとスマートで簡単な単語帳をつくる方法もあります。
英単語とその意味が各ファイルの中に入っています。その中からとり出した英単語の意味を問うものと、その逆である英単語に直しなさいという2通りの問題を出します。
プログラムでは、英単語、その意味の入っているファイルとして一次元のストリング配列を用いることにします。
さあ、つぎのプログラムをRUNさせると、いろいろと問題がでてきます。あなたの英語力をちょっと試してみてください。






```
10  DIM  A$(10), B$(10), C$(10)
20  FOR  J=1  TO  10
30  READ  A$(J), B$(J)
40  C$(J)=A$(J)+B$(J)
50  NEXT  J
60  K=INT(10*RND(1))+1
70  PRINT"[C]タンゴ゛ ノ  イミ  ハ  ナニ?"
80  PRINT  A$(K),
90   INPUT  X$
100  AX$=A$(K)+X$
110  IF  C$(K)=AX$  THEN  PRINT" OK!!":GOTO  150
120  PRINT"マチガ゛イ":FOR  M=1  TO  500:NEXT  M
130  PRINT"[↑]";SPC(10):PRINT"[↑][↑]";TAB(12);SPC(25)
140  PRINT"[H]":GOTO  80
150  K=INT(10*RND(1))+1
160  PRINT"[C]エイゴ゛ニ  ナオシナサイ"
170  PRINT  B$(K),
180   INPUT  Y$
190  YB$=Y$+B$(K)
200  IF  C$(K)=YB$  THEN  PRINT" OK!!":GOTO  60
210  PRINT"マチガ゛イ":FOR  M=1  TO  500:NEXT  M
220  PRINT"[↑]";SPC(10):PRINT"[↑][↑]";TAB(12);SPC(25)
230  PRINT"[H]":GOTO  170
240  END
250  DATA  ATOM,ゲ゛ンシ,BOY,ショウネン,CAT,ネコ,DISH
260  DATA  サラ,ELEPHANT,ゾ゛ウ,FISH,サカナ,GIANT
270  DATA  キョジ゛ン,HORSE,ウマ,ROSE,バ゛ラ,TEACHER
280  DATA  センセイ
RUN
タンゴ゛ノ  イミ  ハ  ナニ?
FISH        ?
```

RND( )は74ページ

この場合、FISHの意味をきいているわけですから、サカナと入れればOK!!と表示して正解であることをあなたに伝えてくれます。それ以外の答には、マチガイと表示します。さらに、サカナときいてきて、あなたがFISHと答える場合もあります。

２次元配列は、さらに強力

学生３人の数学と英語のテストの結果の表（下図の左）にすこし手をくわえた表を作ってみました（下図の右）。

科目＼学生	中　北	谷　町	藤　本
数　学	92	75	72
英　語	70	94	78
国　語	65	60	95

	学　生	中　北	谷　町	藤　本
科　目	M＼N	1	2	3
数　学	1	A(1,1)	A(1,2)	A(1,3)
英　語	2	A(2,1)	A(2,2)	A(2,3)
国　語	3	A(3,1)	A(3,2)	A(3,3)

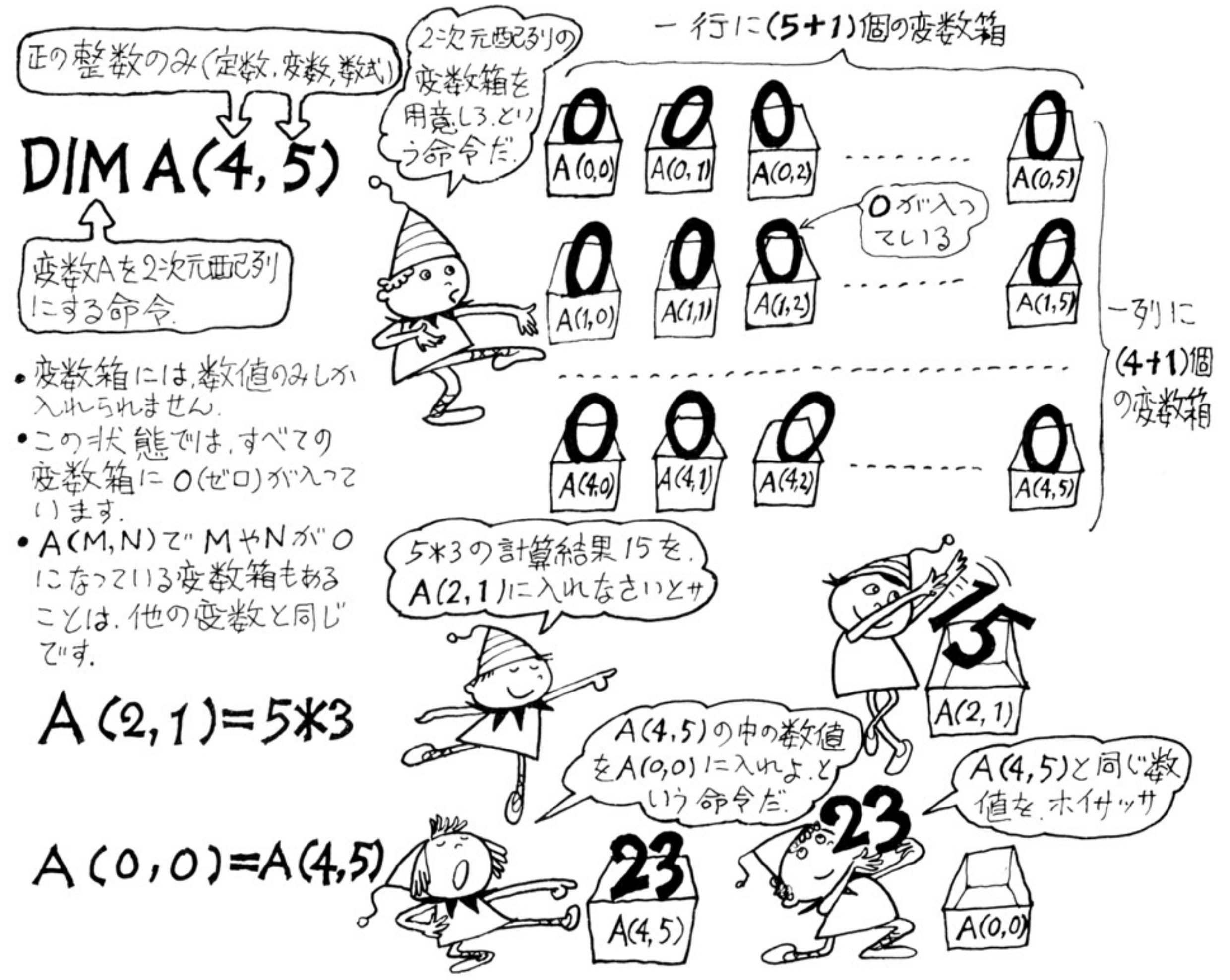
M＝1……数学
M＝2……英語
M＝3……国語

N＝1……中北
N＝2……谷町
N＝3……藤本

右の表では科目をM（１～３）、学生をN（１～３）として、それぞれの点数を A（M，N）という表現で示しています。このようにすると、とても便利なことがあります。たとえば

A（２，２）→M＝２だから英語、N＝２だから谷町君→谷町君の英語の点数

どうです。A（２，２）と書いただけで谷町君の英語の点数であることがすぐにわかります。A（M，N）のMとNはちがった項目をさしていたわけですね。このように２つの項目をつかって、A(M，N)と書いたものを２次元配列といいます。項目が２つあるので２次元というのです。前に出てきた一次元配列はこの項目が１つだけだったのです。
コンピュータでこの２次元配列を使うときどのようにすればよいかをつぎにみてみます。



九九の計算表はいかが

あなたの子供さんはもう九九はできますか。えっ、子供がいるほど歳とっちゃいないって?いずれ将来必要になったときは使ってください。
プログラムには2次元配列A（M, N）を用いることにします。つまりM×Nの値をA（M, N）に割り当てるわけです。




```
10  DIM  A(9, 9)
20  PRINT  "C"
30  FOR  J=1  TO  79
40  SET  J, 6  :NEXT  J
50  FOR  J=3  TO  27
60  SET  5, J:NEXT  J
70  PRINT  "Hクク  ノ  ケイサン"
80  PRINT
90  FOR  J=1  TO  9
100  PRINT  TAB(4*(J-1)+4); J;
110  NEXT  J:PRINT:PRINT
120  FOR  M=1  TO  9
130  PRINT  M;
140  FOR  N=1  TO  9
150  A(M, N)=M*N
160  PRINT  TAB(4*(N-1)+4);
170  PRINT  A(M, N);
180  NEXT  N
190  PRINT
200  NEXT  M
210  END
RUN
クク  ノ  ケイサン
```

	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

```
READY
```

上の表を作るのにわざわざ、2次元配列に数値を代入する必要はないのですが、いったん配列変数A（M, N）に代入しておけば、任意のM, Nに対して、九九の値が計算なしで呼び出せることになります。2次元配列は、データの行列（マトリックス）を構成し、データの取扱いを2次元的に行える特徴があるのです。

ストリング変数にも2次元配列を使用することができます。 数値変数の2次元配列と同様の考え方でストリングの2次元マトリックスを構成することができ、簡単な例として、1学年の生徒の名前を、クラス番号と、出席番号の2次元データとしてまとめた名簿を作成する場合など便利だと思われます。

大きな数と小さな数

花子「ネエ、太郎君。あなた、もしいくらお金使ってもいいといわれたら、何を買う？」

太郎「ウーム、いくら使ってもいいなら……地球の陸地を全部買い占めてやろうかな、土地は資本だというからね。」

花子「あきれた。そんなこと不可能でしょ。もう少し現実的な話をしてよ。」

太郎「それじゃまあ日本中の土地でいいや。」

花子「どっちにしろ、できっこないでしょ。第一日本の土地は、全部で372,000km^2もあるのよ。仮りに１m^2当たり１万円として計算しても3,720,000,000,000,000円にもなるわ」

太郎「そんなに０が並ぶことになるわけ？」

花子「三千七百二十兆円。１万円札にして、１秒間に10枚ずつ数えても、約1,180年かかるお値段なの。あなた、数えてみる？」

太郎「ワー、金持になるのも楽じゃないよこれじゃ。土地の買い占めはよした。だけど、０がずらーっと並んだときの読み方っていうのは、兆や京までじゃないんだって？」

花子「日本の数の読み方は、インドから中国を通って伝わったのを、江戸時代の数学者吉田光由(よしだみつよし)が塵劫記(じんこうき)という本の中で、４けたおきに規則正しく整理して発表したのが一番有名よ。それによると10^{76}という大きな数まで読むことができるの」

太郎「京、垓、秭………那由他、不可思議……全く不可思議な名前だなあ。千大数なんてのは、いったいどのくらいの数になるんだい？」

花子「宇宙にある原子の数が３×10^{74}というから、ちょうどそれに相当するぐらいの数よ。」

太郎「フーン、全然想像つかないのに変わりはないけれどスゴイね。」

花子「インドの古い文献によると10^{140}まであらわすことができるのよ。お釈迦(しゃか)様なんかが、そん

指数と仮数



$.12E+04 = 0.12 \times 10^4 = 1200$

.12 … 仮数　04 … 指数



□□□□, □□□□, □□□□, □□□□, □□□□, □□□□, □□□□, □□□□, □□□□, □□□□, □□□□, □□□□,

千	百	十	一
千大数	百大数	十大数	大数(たいすう)
千無量	百無量	十無量	無量(むりょう)
千不可思議	百不可思議	十不可思議	不可思議(ふかしぎ)
千那由他	百那由他	十那由他	那由他(なゆた)
千阿僧祇	百阿僧祇	十阿僧祇	阿僧祇(あそうぎ)
千恒河沙	百恒河沙	十恒沙	恒河沙(こうがしゃ)
千極	百極	十極	極(ごく)
千載	百載	十載	載(さい)
千正	百正	十正	正(せい)
千澗	百澗	十澗	澗(かん)
千溝	百溝	十溝	溝(こう)
千穣	百穣	十穣	穣(じょう)

人間(男)が70年間に吸い込む空気の量 ------------ .1325E+09 ℓ

新聞紙を半分に割って重ね,それを半分に割って重ね…という風に40回繰り返して重ねると,その厚さは ------------ .165E+12 mm

ただし,そのときの一枚の紙の大きさは ------ (.78 E−03)X(.52E−03) mm

1年間に世界中に降る降水量 ------------ .41E+15 ton

地球の海水の量 ------------ .14E+19 ton

地球の質量 ------------ .5977E+22 ton

太陽の質量 ------------ .1991E+28 ton

小さな数と大きな数

な大きな数まで使って、いろいろと哲学的なことなんかを考えているんでしょうね」

太郎「小さい数の方はどう？」

花子「小数のお話も、やっぱりお釈迦様の問答の中に出てくるわよ。１けたごとに全部名前がついてるの。それによると、小数以下２３けたまでの小さな数を言いあらわすことが可能ね」

太郎「これもまた奇妙なのが並んでるけど、塵とか埃とか、模糊、刹那なんて、感じ出てるね。」

花子「水素原子の直径は約１オングストロームというけど、㎞単位だと１模糊㎞、㎝単位だと１沙㎝ということになるのかしら。」

太郎「フムフム……ところでボクらのコンピューターは、こういう大きい数や小さい数も得意なんだろうね。」

花子「デ－タの表現次第よ。もちろん無限ではないけれどね。たとえば8ビットといったら、８桁の２進数だから２５６の数が表現できるし、４桁の１６進数といったら、６５５３６の数が表現できることになる。」

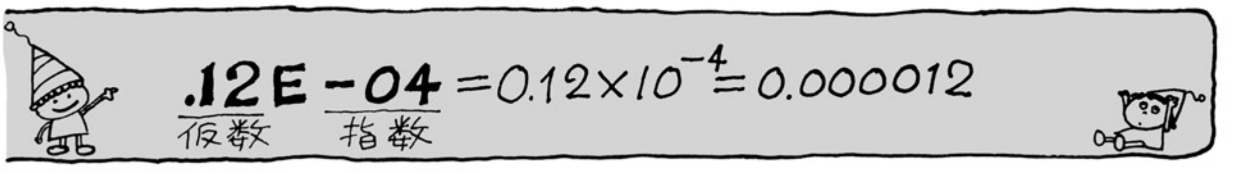
太郎「２進数に１６進数か………」

花子「いくらコンピュータでも整数や小数をそのまま扱うんじゃとてもかなわないのよ。そこで浮動小数点方式といって、数の有効数字を決めてそれに指数部を加えた表現を使うことによって、扱える数の範囲を広げているわけなのよ。」

太郎「電卓でＥＸＰ部のついたのを見たことあるけど、あれもその指数表現のためのものだね。」

花子「そうよ。ＢＡＳＩＣでは有効数字８桁として、$0.27105055 \times 10^{-19}$から$0.92233720 \times 10^{19}$までの数が扱えるわ。」

太郎「へ－、ずいぶんよく勉強してるんだなあ。いくらでもお金が使えるのもいいけれど、せめて10^3倍ぐらい頭がよくなりたい気がするよ。」



□□□□，□□□□，□□□□，□□□□，□□□□，□□□□，□□□□．□□□□□□□□□□□□□□□□□□□□□□□

千秭 百秭 十秭 秭(じょ)，千垓 百垓 十垓 垓(がい)，千京 百京 十京 京(けい)，千兆 百兆 十兆 兆(ちょう)，千億 百億 十億 億(おく)，千万 百万 十万 万(まん)，千 百 十 一．

分(ぶん) 厘(りん) 毫(毛)(もう) 絲(糸)(し) 忽(こつ) 微(び) 纖(せん) 沙(しゃ) 塵(じん) 埃(あい) 渺(びょう) 漠(ばく) 模糊(もこ) 逡巡(しゅんじゅん) 須臾(しゅゆ) 瞬息(しゅんそく) 弾指(だんし) 刹那(せつな) 六徳(りっとく) 虚(きょ) 空(くう) 清(せい) 浄(じょう)

銀河宇宙の直径 ………………………… .95E＋18 km

〃　　の総質量 ………………………… .4E＋42 kg

原子(Na)の半径 ………………………… .191E－11 m

10,000円/m² としたときの地球陸地のお値段 ………… .149E＋11 円

人間の心臓が70年間に送り出す血液の量 ………… .184E＋09 l



乱数はデタラメな数



プロ野球のピッチャーが乱数表というものを使っているのを知っていますか。乱数表にあるような乱数列というのは、どの数も発生する割合が等しく、さらに数の発生する仕方に規則性のまったくない（ランダムといいます）数の列をいいます。そのときの数は乱数といわれます。

0,1,2,3,4,5,6,7,8,9,0,1,2,3,……

この数列はどの数も発生する割合は等しいですね。しかし、発生の仕方がランダムではありません。つまり、乱数ではないのです。

0,1,1,1,8,7,3,9,1,6,1,2,1,1,5, …

この数列では発生の仕方はかなりランダムですが、1の発生する割合が非常に大きくなっています。ですから、これも乱数とはいえません。

あなたも乱数列を考えてみませんか……。意外とむずかしいことがおわかりでしょう。でも心配はいりません。コンピュータが、この乱数をつくってくれます。RND(　)を用いればよいのです。(　)の中にはあなたの好きな正の整数を入れてください。なんでもよいのです。

```
10 FOR I=1 TO 3
20 PRINT RND(2);:NEXT:END
RUN
 0.38079483 0.75828109 0.44046507
READY
```

乱数だから結果はこのようになるとはかぎりません。

結果をみるとわかると思いますがRNDのとる値にはつぎの関係があります。

0<RND（1）<1

0と1を含んでいないことに注目

1000個の乱数の平均をとってみます。

```
10 FOR J=1 TO 1000
20 R=RND(1):S=S+R
30 NEXT J:RG=S/1000
40 PRINT RG
50 END
RUN
 0.4980582
READY
```

この平均をみてみると、0.5に非常に近い値となっていますね。このことからも、ここで発生した乱数がランダムであることがわかります。

RND関数でサイコロをつくる



サイコロを振ってみてください。当然のことですが、１から６までのどれかの目が出ますね。サイコロのもつおもしろさはどの目も出る割合が同じであるということです。つまり１から６までの目がランダムに発生するわけです。
コンピュータには乱数を発生させるRND関数がありますが、この関数を用いてサイコロを作れないかということです。じつはできるのです。ではどうするのでしょう。まずはRND関数を６倍してください。

$$0<\mathrm{RND}(1)<1 \xrightarrow{\times 6} 0<6\times\mathrm{RND}(1)<6$$

ここでＩＮＴ命令を思い出してください。６×RND(１)にＩＮＴ命令を用いるのです。

```
INT (6*RND(1))＝0, 1, 2, 3, 4, 5
```

これで０から５までのランダムに発生する整数を作ることができました。サイコロは１から６までですから、まだだめです。さらに１をたせばよいと思いつきましたか。

```
INT (6*RND(1)) +1＝1, 2, 3, 4, 5, 6
```

さあ、サイコロが一丁できあがりです。さっそくこのサイコロを使って、目の出る割合を調べてみませんか。




```
10  PRINT  "サイコロ  ヲ  フル  カイスウ";
20  INPUT  N
30  FOR  J=1  TO  N
40  R=INT(6*RND(1))+1
50  IF  R=1  THEN  N1=N1+1
60  IF  R=2  THEN  N2=N2+1
70  IF  R=3  THEN  N3=N3+1
80  IF  R=4  THEN  N4=N4+1
90  IF  R=5  THEN  N5=N5+1
100  IF  R=6  THEN  N6=N6+1
110  NEXT  J
120  P1=N1/N:P2=N2/N:P3=N3/N
130  P4=N4/N:P5=N5/N:P6=N6/N
140  PRINT  P1, P2, P3, P4, P5, P6
150  END
RUN
サイコロ  ヲ  フル  カイスウ?  5000
  0.1702      0.1654      0.1628      0.1626
  0.169       0.17
READY
```

どうです。どの目も出る割合はほとんど同じでしょう。数学的にいえば、どの目も６回振って１回だけでるサイコロが理想的なわけで、そうすれば1/6つまり０．１６７ に近いほどいい乱数サイコロといえるのです。

算数の家庭教師に早がわり

あなたの小さい弟、妹クンたちのために算数の先生をつくってあげましょう。はじめに1から9までの数の足し算とかけ算の練習から。足し算の練習は……



```
10  A=INT(RND(1)*9+1)
20  B=INT(RND(1)*9+1)
30  PRINT"  ";A:PRINT"+";B
40  PRINT"□□□□□□":INPUT C
50  IF C=A+B THEN 80
60  PRINT"モウイチト゛ヨクカンカ゛エテ"
70  GOTO 30
80  PRINT"OK! ヨクテ゛キマシタ":GOTO10
```

文番号10と20で乱数を使って1から9までの2つの数を変数AとBに代入します。つぎにその2つの数の足し算の式を書いて、答えを「?」ときいてきます。そして文番号50で答えが正しいかどうか判定するのです。
かけ算の練習は文番号30と50の"+"記号を"*"に変えればいいのです。
つぎは1から99までの数です。文番号10と20を




```
10  A=INT(RND(1)*99+1)
20  B=INT(RND(1)*99+1)
```

とします。たしかにこれでよさそうです。ところが1ケタの数が出てくると2つの数のケタが合わなくなりますね。(右図) そこでケタを合わせるように細工します。つまりストリング処理を使って、数値のケタ数を調べ、それにしたがって式を書くわけです。前のプログラムに更につぎの変更を追加すれば完成です。

```
25  A$=STR$(A):B$=STR$(B)
30  PRINT TAB(5-LEN(A$));A
35  PRINT "+";TAB(5-LEN(B$));B
40  PRINT "□□□□□□":INPUT C
```

文番号25で2つの数をストリングに直し、文番号30と40とでそれぞれの数を、そのケタ数に合わせて表示します。TAB( )とLEN( )の使い方に注目してください。

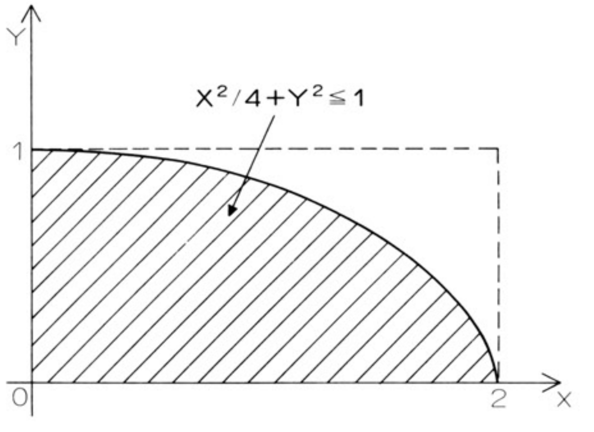
ティータイム

勉強を中断して、ティータイムにしましょう。
弟や妹クンたちと、乱数模様を見ながらケーキを食べるのもステキですね。

```
100  ?"[C]";
110  ?"[H]":W=1
120  FOR X=1TO7:FOR Y=1TO5:FOR Z=1TO5
130  ? TAB((X-1)*5);A$;:NEXT:?:NEXT
140  A$=CHR$(INT(RND(1)*223+33))
150  ?"↑↑↑↑↑";:NEXT:?"↓↓↓↓":W=W+1
160  IF W>4 THEN 110
170  GOTO 120
```

確率的に図形の面積を求める

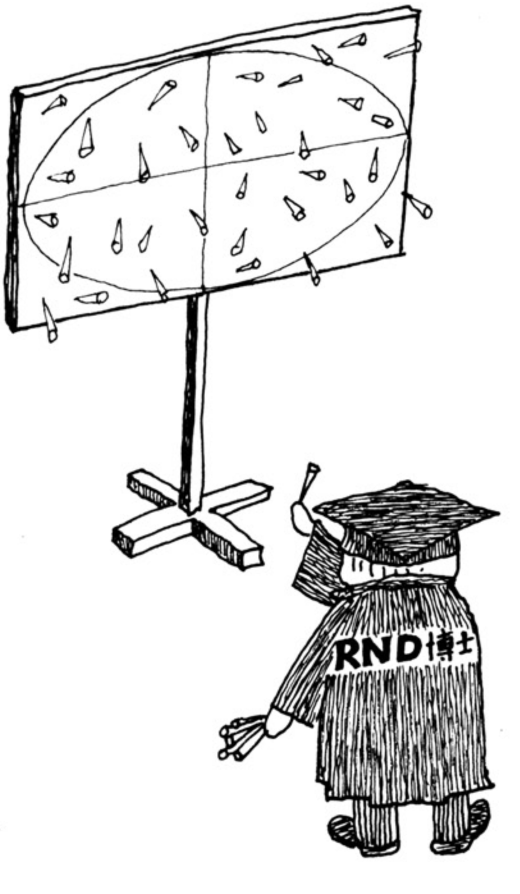
乱数を用いると積分を行なうことができます。この場合、確率を用いますが、これはモンテカルロ法による数値積分といわれるものです。
だ円の面積（定積分）を求めてみましょう。簡単のため4分の1のだ円を考えます。右の図で斜線部分は $X^2/4+Y^2=1$ のだ円内部です。いま点線でつくられる長方形にランダムに矢を投げることにします。矢を数多く投げれば、だ円内部に当った回数と矢を投げた総回数の比はだ円の面積と長方形の面積の比に近づいてゆくことになります。
投げ矢の役をRND関数にやってもらうことにします。だ円の面積を求めるプログラムはつぎのようになります。




```
10 PRINT "ヤ ヲ ナゲ゛ル カイスウ ハ ?"
20 INPUT N
30 PRINT "ケイサンチュウ デ゛ス"
40 FOR J=1 TO N
50 X=2*RND(1):Y=RND(1)
60 IF X*X/4+Y*Y<=1 THEN ND=ND+1
70 NEXT J
80 S=4*(2*ND/N)
90 PRINT "ダ゛エン ノ メンセキ S=";S
100 END

RUN
ヤ ヲ ナゲ゛ル カイスウ ハ ?
? 100
ケイサンチュウ デ゛ス
ダ゛エン ノ メンセキ S= 6.4
READY

RUN
ヤ ヲ ナゲ゛ル カイスウ ハ ?
? 1000
ケイサンチュウ デ゛ス
ダ゛エン ノ メンセキ S= 6.336
READY
```



このだ円の実際の面積は約6.28です。矢を100回投げたとき、6.4ですからかなり近い値になっています。回数を1000としたときには、さらに実際の面積に近くなっています。でたらめな数を使ってこうして面積を求めるなんておもしろいと思いませんか。しかも数がでたらめならでたらめなほど、答は正確になって行くのです。

カジノでかせごう $$スロットマシン$$

こちらラスベガス。歓楽と一獲千金の夢のるつぼの街。ギャンブル好きのマイケルはスロットマシンで遊んでいます。10セント硬貨を入れてバーをガチャンと押します。ガラガラと回転して、三つの窓に◆◆◆（ダイヤ）とか♠♠♠（スペード）とか出てきて、その組み合わせによってコインがジャラジャラ出てきます。
もしはずれれば10セントをなくします。さあプログラムを作ってみましょう。

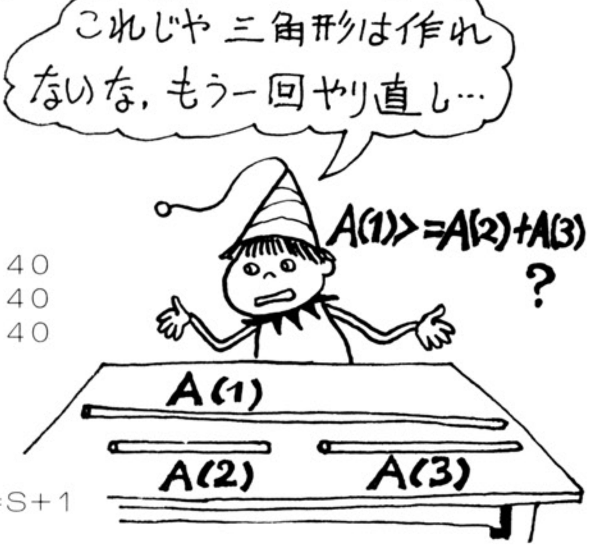



```
10 PRINT"[C]":D=0:H$="[H]↓↓↓↓↓↓↓↓"
20 DIM N$(5),A(3)
30 FOR X=1 TO 10:H$=H$+"↓→":NEXT X
40 FOR X=1 TO 5:READ N$(X):NEXT X
50 PRINT H$;"↑↑↑$=          ←←←←←←";D
60 PRINT H$;"↓↓":PRINT SPC(38);"↑"
70 INPUT "バーヲ オスチカラハ ?"; NN
80 FOR Y=0 TO NN:RR=RND(1):NEXT Y
90 FOR X=1 TO 3:A(X)=INT(5*RND(1))+1:NEXT X
100 PRINT"[H]↓↓↓";
110 FOR X=1 TO 3:PRINT N$(A(X)),:NEXT:PRINT
120 IF A(1)<>A(2) THEN 170
130 IF A(2)<>A(3) THEN 160
140 IF A(1)=5 THEN PRINT H$;"ダイショウリデス!!
    $1.00    ":D=D+1.00:GOTO 50
150 PRINT H$;"ミンナオナジデス!$.25    ":D=D+0.25
    :GOTO 50
160 PRINT H$;"ショウキン ハ $.10     ":D=D+0.1
    :GOTO 50
170 IF A(1)=1 THEN PRINT H$;"ショウキン ハ $.05
        ":D=D+0.05:GOTO 50
180 PRINT H$;"ザンネン. モウ 10セント ドウゾ          "
    :D=D-0.1: GOTO 50
190 DATA ♠♠♠,◆◆◆,♥♥♥,♣♣♣,$$$
```

うまく行きましたか。え?………コインがジャラジャラなんてウソばっかりって?　まあそこのところは………。マイケルと同様ギャンブル好きのきみ（?）、もうちょっと凝って、たとえばスロットマシンの形なども作ってみたくなったんじゃないかな！

RNDで練習問題をつくろう

三角形の面積を求める練習問題をRND関数につくってもらいましょう。ここではRND関数があなたの先生です。乱数によって三角形の辺の値をいろいろと与えてくれますから、あなたはその面積を計算して答えます。RNDはあなたのよきパートナーです。

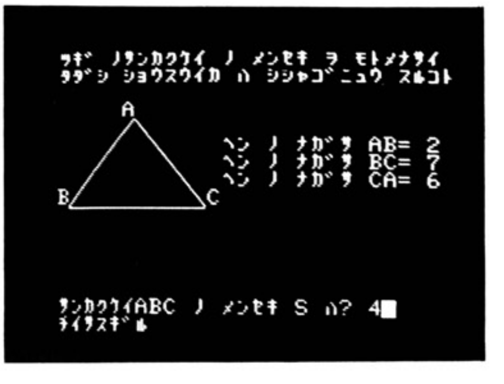



```
10  DIM A(3),L$(4)
20  FOR J=1 TO 4
30  READ L$(J) :NEXT J
40  FOR J=1 TO 3
50  A(J)=INT(20*RND(1))+1
60  NEXT J
70  IF A(1)>=A(2)+A(3) GOTO 40
80  IF A(2)>=A(1)+A(3) GOTO 40
90  IF A(3)>=A(1)+A(2) GOTO 40
100 W=(A(1)+A(2)+A(3))/2
110 T=W :FOR J=1 TO 3
120 T=T*(W-A(J)) :NEXT J
130 SS=SQR(T) :S=INT(SS)
140 IF SS-S>=0.5 THEN S=S+1
150 PRINT "C↓↓↓↓"
160 PRINT "    ツギ゛ ノ サンカクケイ ノ メンセキ ヲ モトメナサイ"
170 PRINT "    タダ゛シ ショウスウイカ ハ シシャコ゛ニュウ スルコト"
180 PRINT
190 PRINT TAB(8);"A"
200 PRINT TAB(8);"／＼"          ;TAB(15);L$(1);A(1)
210 PRINT TAB(7);"／  ＼"        ;TAB(15);L$(2);A(2)
220 PRINT TAB(6);"／    ＼"      ;TAB(15);L$(3);A(3)
230 PRINT TAB(5);"／      ＼"
240 PRINT TAB(3);"B／        ＼C"
250 PRINT TAB(4);"□□□□□□□□□□□"
260 PRINT "↓↓↓"
270 PRINT TAB(3);L$(4);
280 INPUT Y
290 IF Y=S  THEN PRINT "       OK!!":GOTO 40
300 IF Y<S  THEN PRINT "    チイサスギ゛ル":GOTO 320
310 PRINT "    オオキスギ゛ル"
320 PRINT "↑↑";
330 PRINT TAB(24); SPC(25) :PRINT "↑";
340 GOTO 270
350 DATA ヘン ノ ナガ゛サ AB=,ヘン ノ ナガ゛サ BC=
360 DATA ヘン ノ ナガ゛サ CA=,サンカクケイABC ノ メンセキ S ハ
```

試して欲しいこと！

RND( )の( )の中に0または負の整数をいれると、乱数値はある一定の値しか得られません。これは初期値をセットしているのです。





SETする？ それともRESETする？

ためしにダイレクトモードで、つぎの命令を与えて下さい。

SET　79，49 [CR]

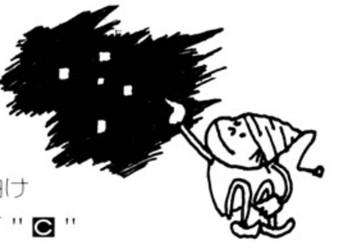


テレビ画面

どうでした？　テレビ画面の右下隅が光っているはず。
そこでこんどは

RESET　79，49 [CR]

点灯しているところが消えてしまいましたね。
実はSET，RESET命令は、右図のようになる一点だけを光らせたり、消したりする働きを持っているのです。



前方右下隅に点滅物を発見！

```
10 PRINT"■"
20 X=79:Y=49
30 SET X,Y      ← 光る
40 RESET X,Y ← 消える
50 GOTO 30
```

画面大の長方形を書くために

```
10 PRINT"■"
20 FOR X=0 TO 79
30 SET X,0
40 SET X.49
50 NEXT X
60 FOR Y=0 TO 49
70 SET 0,Y
80 SET 79,Y
90 NEXT Y
99 GOTO 99
```

テレビ画面はまっ白け

```
10 PRINT"■"
20 FOR X=0 TO 79
30 FOR Y=0 TO 49
40 SET X,Y
50 NEXT Y,X
60 GOTO 10
```

斜めの線を書くために

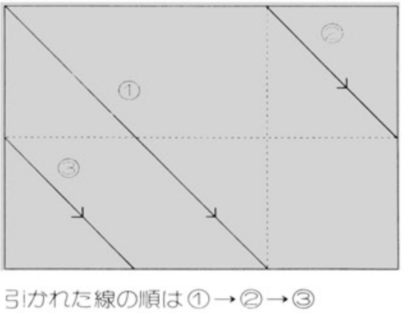
```
10 PRINT"■"
20 PRINT"ナナメノナガサ";
30 INPUT Y
40 X=SQR(Y*Y/2)  ←
50 FOR A=1 TO X
60 SET A,49-A
70 NEXT A
80 GOTO 80
```

三平方の定理を応用してます。

ここでちょっと変わったことをしてみましょう。

```
10 PRINT"■"
20 FOR Z=1 TO 99
30 SET Z,Z
40 NEXT Z
50 GOTO 50
```

実行したあとのテレビ画面



引かれた線の順は①→②→③

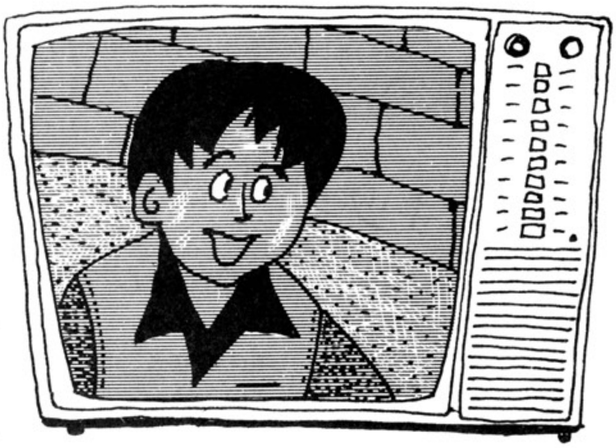
そうなんです。実はX方向で８０以上、Y方向で５０以上になったら

X ⟵ X－80
Y ⟵ Y－50

が自動的に計算されるしくみなんですね。

テレビの原理を紹介すると…

```
10  PRINT"🅲":Y=1
20  FOR X=0 TO 79
30  SET X,Y
40  RESET X,Y-1
50  NEXT X
60  Y=Y+1
70  GOTO 20
```



縦へひとつだけずれながら、横へ線が光って走るのが見えたでしょうか、時間をできるだけ引き伸ばして原理がわかるようにしたものです。
本物のテレビの場合、このような線約500本で1枚の画面ができ、これが1秒間に30枚の割で送られてきているのです。

ボールを飛びはねさせると…

```
10  PRINT"🅲"
20  FOR X=0 TO 79
30  SET X,0:SET X,49
40  NEXT X
50  FOR Y=0 TO 49
60  SET 0,Y:SET 79,Y
70  NEXT Y
80  X=79*RND(1):Y=49*
    RND(1)
90  A=1:B=1
100  SET X,Y
110  IF X<2 GOSUB 200
120  IF X>78 GOSUB 200
130  IF Y<2 GOSUB 250
140  IF Y>48 GOSUB 250
150  RESET X,Y
160  X=X+A:Y=Y+B:GOTO 100
200  A=-A:MUSIC"□A0":
     RETURN
250  B=-B:MUSIC"A0":
     RETURN
```

どうです。プログラムの流れがつかめましたか?
このプログラムは3つのポイントがあります。ひとつはボールの出発点を乱数で作り出す文番号80、ふたつ目はボールが四方の壁にぶつかったかどうかを調べる文番号110~140、三つ目は壁にぶつかったとき方向転換するための文番号200と250。

池に小石を投げてみると……

```
10  X=40:Y=25
20  DEF FNY(Z)=SQR(R*R
    -Z*Z)
30  PRINT"🅲":SET X,Y
40  R=R+5
50  FOR Z=0 TO R
60  T=FNY(Z)
70  SET X+Z,Y+T
80  SET X+Z,Y-T
90  SET X-Z,Y+T
100 SET X-Z,Y-T
110 NEXT Z
120  IF R<>25 THEN 40
130 GOTO 130
```



野生のスケッチ ウサギとキツネ

動物の生態に関するひとつのモデルを、コンピュータのSET命令を使ったグラフィックで見てみることにしましょう。ここでは動物の生存競争のモデルを取り上げてみます。生態学によってこのモデルは微分方程式の形で示されていますが、コンピュータによる数値計算の簡単な例としても参考になると思います。

ウサギとキツネ

動物の世界で言われる弱肉強食の関係としてウサギとキツネを例に取り上げましょう。そしてキツネは草食のウサギを常食とするものとします。
生態学のモデルによるとウサギの数をX、キツネの数をYとすると、それぞれの個体の数は、つぎの微分方程式にしたがう変化をしていきます。

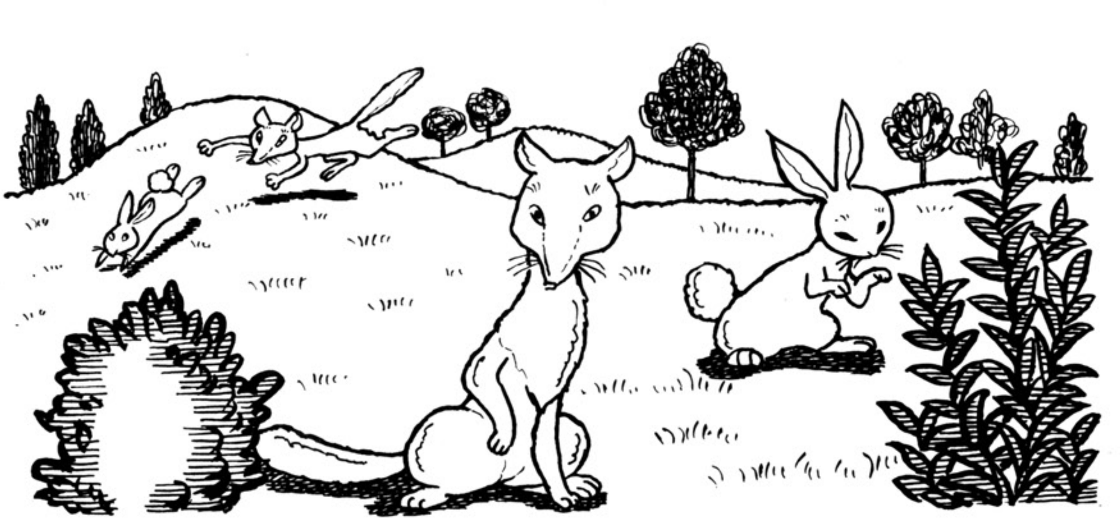
$\frac{dx}{dt}=AX-BXY$ ………………ウサギの数の変化

$\frac{dy}{dt}=CXY-DY$ ………………キツネの数の変化

A、B、C、Dは適当な定数。XYの項がウサギとキツネの弱肉強食の意味をもっています。ウサギの数の変化を見てみますと、もしキツネがいなかったら第二項はゼロとなり、ネズミ算的に繁殖して行きますが、キツネがいるとキツネの数も増加しウサギの数の増加はおさえられることになります(第二項)。
キツネの数の変化では、ウサギがいなかったら第一項はゼロとなり、キツネはどんどん死んでいきます。
ウサギがいるとキツネの数は第一項のCXYの値にしたがって増加していく、という関係になっています。
この微分方程式をコンピュータで数値的に見るために、「接線法」による近似を採用します。それは時間微分dtを有限な微小時間DTで近似するものでこのときdx、dyをDX、DYとすると、この微分方程式は

DX=(AX−BXY)DT
DY=(CXY−DY)DT

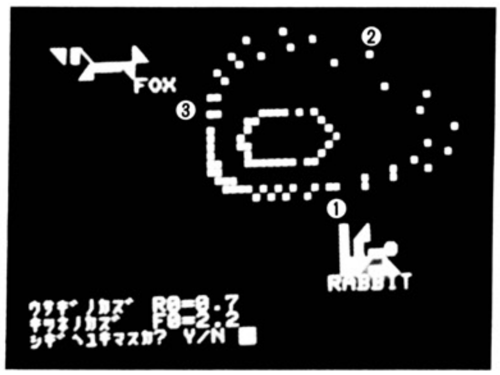
と代数方程式になります。
つぎのページにこの方法を用いたプログラムの例を示します。



卵形のグラフの秘密 ウサギとキツネ

```
10 REM ウサギト キツネ
20 READ A,B,C,D,DT,TT,K
30 DATA 4,2,1,3,0.01,2,30
40 FOR H=1 TO 20:H$=H$+"⬇":NEXT H
50 PRINT"🅲⬇⬇⬇⬇⬇⬇⬇⬇➡◥◨◹    ◥":PRINT"➡➡➡◢▬▬◥"
60 PRINT TAB(4);"FOX";"🅷";H$;"⬆⬆⬆";TAB(22);"◨◥"
70 PRINT TAB(22);"◨◨▄●":PRINT TAB(22);"■◢▄◥"
80 PRINT TAB(21);"RABBIT"
100 PRINT"🅷⬇";H$;SPC(38):PRINT SPC(38);"🅷";H$
110 INPUT"ウサギノ カズ R0=";X
120 INPUT"キツネ ノ カズ F0=";Y
200 FOR I=1 TO K
210 FOR T=0 TO TT/K STEP DT
220 X=X+(A*X-B*X*Y)*DT
230 Y=Y+(C*X*Y-D*Y)*DT
240 NEXT T
250 SET 4*(X+6),4*(8-Y)
260 NEXT I
270 INPUT"ツギヘ ユキマスカ? Y/N ";Y$
280 PRINT"⬆";SPC(38);"⬆"
290 IF Y$="Y" THEN 100
300 GOTO 270
```

REMは何も実行せず、ただプログラムの注釈を示します。



さてRUNさせてみましょう。ウサギの数、キツネの数の初期値をきいてきますね。たとえばR0=5、F0=5をキー入力してみてください。この初期値からはじめて、時間とともに数が変化してゆくのが見られます。

きれいな卵形の図形が描かれます。さて、図形の意味を考えてみましょう。

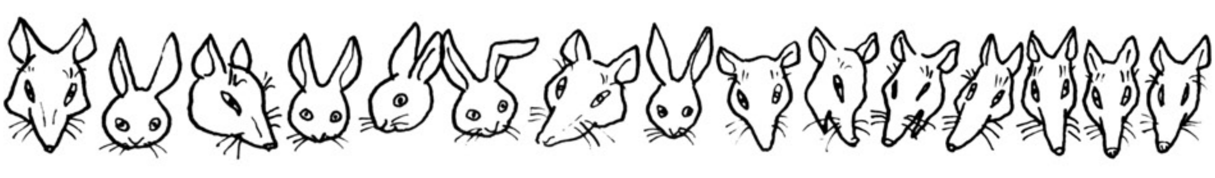
❶の部分。最初キツネの数が少ない時ウサギはどんどん増えて行き、キツネの数もすこしずつ増えて行きます。

❷の部分。キツネの数が増えウサギの数が頭うちとなって逆に減りはじめます。

❸ウサギの数がどんどん減少して行くと、キツネの数も減りはじめ、急激に減少して行きます。

こうしてもとの状態にもどります。ウサギの数とキツネの数の初期設定をいろいろ変えてみるといろいろな大きさの卵形のグラフが得られます。

このプログラムは、ダートマス大学教授で、BASICの開発者であるケメニー、カーツ著の「ベーシック入門」(森口繁一監訳 共立出版)で取り上げられている問題によっています。同書および、ケメニー、スネル、トンプソン共著の「新しい数学」(矢野健太郎訳 共立出版)には、多くの社会、人文科学領域での興味深い数学的問題が提示されています。



GETは待ったなしのキー入力

キーボードからデータ入力する命令の新顔をここで紹介します。その名も

GET



ゲームなどで素早い処理を実行するのにぴったりのヤツです。INPUT命令では最初「?」ときいてきて、データを入れてから CR キーをたたいてはじめて次へ移りますが、GET命令では

◆プログラムの実行がGET命令にあるとき、1つのキーが押されていたら、その1文字分のデータを変数に代入し先へ進む。

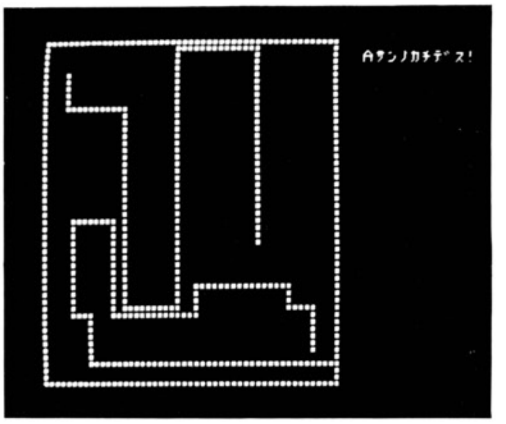
◆もしどのキーも押されていないと数値変数なら「0」、ストリング変数なら「空」が代入され、先へ進む。

となっているのです。
GET命令を用いるときはGET命令の前後をぐるぐる廻りながら、データがキー入力されるのを待っているような使い方が多くなります。
GET命令を使ってゲームをやる前に簡単な練習をしてみましょう。GOTO命令でくり返しGET命令を実行させます。変数Zに対して1から9までの数値をキー入力してみてください。

```
10  GET  Z
20  PRINT  Z;:GOTO  10
```

キーを押せばそのキーの数値が表示されますが、押したままにすると、次からはその数値は入力されず0が表示されます。

さてGET命令、SET命令を使って「陣取りゲーム」を作ってみることにしましょう。プログラムは右ページ。2人で遊ぶゲームです。画面の正方形の枠内で2人がそれぞれ左上と右下から出発して下図のキー操作によって向きを変えながら相手の動きを阻止させるように動かします。相手が行きづまったら勝ちです。



Aサン（左上からスタート）

キー	1	2	3	4
	↓	↑	→	←

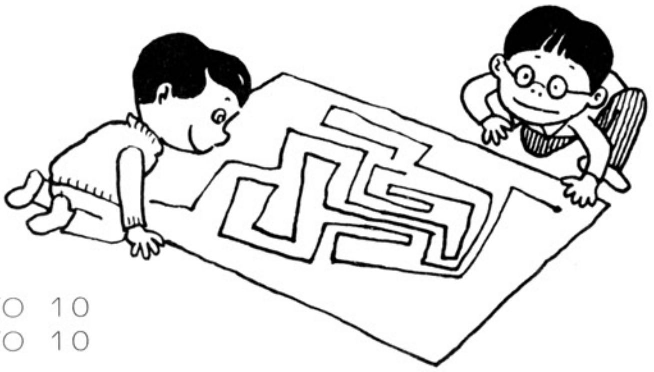
Bサン（右下からスタート）

キー	6	7	8	9
	↓	↑	→	←

ただし、相手がキーを押しているときは、他のキーは受けつけられません。交互にポンと押す感じでキー入力してください。

陣取りゲームをのぞいてみませんか

このプログラムはこれまでの命令をふんだんに使っていますので、どうつくってあるのかのぞいてみたらいかがでしょう。ただしこのプログラムにはほぼ１３キロバイトのプログラムエリアが必要です。



```
10 CLR
20 DIM Z(49,49)
30 PRINT"C";TAB(26);"Aサン・ヒタ゛リウエカラ"
35 PRINT TAB(26);"1  2  3   4"
40 PRINT TAB(26);"シタ ウエ ミキ゛ ヒタ゛リ"
45 PRINT TAB(26);"Bサン・ミキ゛シタカラ"
50 PRINT TAB(26);"6  7  8   9"
55 PRINT TAB(26);"シタ ウエ ミキ゛ ヒタ゛リ"
60 FOR A=0 TO 49
70 SET A,0:Z(A,0)=-1
80 SET A,49:Z(A,49)=-1
90 NEXT A
100 FOR A=1 TO 49
110 SET 0,A:Z(0,A)=-1
120 SET 49,A:Z(49,A)=-1
130 NEXT A
140 X1=4:Y1=4:D1=1
150 X2=45:Y2=45:D2=7
160 GOSUB 200:IF M=1 GOTO 10
170 GOSUB 300:IF M=1 GOTO 10
180 GOTO 160
200 GET X
210 IF X=0 THEN 260
220 IF X=1 THEN Y1=Y1+1:GOTO 270
230 IF X=2 THEN Y1=Y1-1:GOTO 270
240 IF X=3 THEN X1=X1+1:GOTO 270
250 IF X=4 THEN X1=X1-1:GOTO 270
260 X=D1:GOTO 220
270 D1=X:IF Z(X1,Y1)=-1 THEN 400
280 MUSIC"□A0":SET X1,Y1:Z(X1,Y1)=-1
290 RETURN
300 GET X
310 IF X=0 THEN 360
320 IF X=6 THEN Y2=Y2+1:GOTO 370
330 IF X=7 THEN Y2=Y2-1:GOTO 370
340 IF X=8 THEN X2=X2+1:GOTO 370
350 IF X=9 THEN X2=X2-1:GOTO 370
360 X=D2:GOTO 320
370 D2=X:IF Z(X2,Y2)=-1 THEN 450
380 MUSIC"A0":SET X2,Y2:Z(X2,Y2)=-1
390 RETURN
400 PRINT TAB(27);"Bサンノカチテ゛ス!"
410 MUSIC"C3DEG□CEG□CCDEG□CEG□CC":M=1:RETURN
450 PRINT TAB(27);"Aサンノカチテ゛ス!"
460 MUSIC"□G3□A□BCDE#FGAB□C□D□#D□E□F□#F□G":M=1
470 RETURN
```

ＴＩ＄は デジタル時計

プログラムの中に時計の機能を持たせるとおもしろいものです。デジタル時計や世界時計だってプログラム次第！
そんな機能をＴＩ＄が果たしてくれます。
あなたの前にあるコンピュータにつぎのようにキー入力して CR キーを押してみてください。

```
PRINT TI$
```



たとえば、つぎのような表示があらわれると思います。

```
001234
```

何だろう

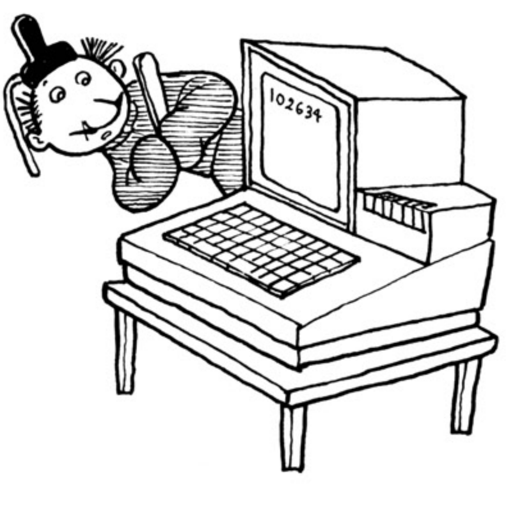
ストリング変数ＴＩ＄の内容であるこの６ケタの数字は何を意味しているのでしょうか？
実は、つぎのように時刻を表わしているのです。

001234 ⇨ ００時１２分３４秒

時 分 秒

ただし、あなたの時計とあっているかどうかはわかりません。その理由はというと、コンピュータの電源を入れると同時にＴＩ＄には００時００分００秒が自動的にセットされるからです。
もう一度、ＴＩ＄を表示させてみてください。前にやったときとＴＩ＄の内容がちがっていることに気づいたと思います。そうです。ＴＩ＄の中の６ケタの数字はデジタル時計の文字盤と全く同じように刻々とかわって行くのです。今までのＴＩ＄の値は、じつは電源を入れたときからの経過時間だったのです。

```
10 TI$="102634"
20 PRINT TI$
30 END
RUN
102634
READY
```



文番号１０のようにするとＴＩ＄にあなたの好きな時刻がセットされます。ただし、"　"の中は６ケタの数字に限ります。

パリの友人に、おはようって電話する時間は？

Ａ＄のようなふつうのストリングとの関係をつぎのプログラムでみてください。




```
10  A$="123456"
20  TI$=A$
30  PRINT  TI$
40  FOR  T=1  TO  1000:NEXT
50  B$=TI$
60  PRINT  B$
70  END
RUN
123456
123457
READY
```

プログラムの実行中にＴＩ＄は１秒経過しています。

簡単な世界時計をつくってみました。もちろんデジタル時計です。あなたの好きな都市の時刻をおしえてくれます。さあ、つぎのプログラムをRUNさせてください。ここでもＴＩ＄が時を刻んでいるのです。ロンドンと東京は9時間もちがっているのですね。こんどは、あなたの手で他の都市の時刻をおしえてくれるプログラムをつくってみてはいかがですか。

```
10  PRINT  "[C]"
20  DIM  C$(10),D(10),E(10),T$(10)
30  FOR  I=1  TO  10:READ  C$(I),D(I)  :NEXT  I
40  PRINT"トーキョウ ノ シ゛コク ハ ?"
50  INPUT  TI$:PRINT"[C]"
60  PRINT  "[H]":T$(1)=TI$
70  FOR  I=1  TO  10
80  E(I)=VAL(LEFT$(T$(1),2))+D(I)
85  IF  E(I)=24  THEN  E(I)=0
90  IF  E(I)<0  THEN  E(I)=24+E(I)
100  T$(I)=STR$(E(I))+RIGHT$(T$(1),4)
110  IF  LEN(T$(I))=5  THEN  T$(I)="0"+T$(I)
120  PRINT  C$(I);TAB(15)  ;LEFT$(T$(I),2);
130  PRINT"時  ";MID$(T$(I),3,2);"分  ";
140  PRINT  RIGHT$(T$(I),2);"秒  ":NEXT  I
150  GOTO  60:END
160  DATA  トーキョウ,0,モスクワ,-6,リオテ゛シ゛ャネイロ,-12
170  DATA  シト゛ニー,1,ホノルル,-19,ロント゛ン,-9,カイロ,-7
180  DATA  ニューヨーク,-14,サンフランシスコ,-17,ハ゜リ,-8
```

時差	都市
-12	リオデジャネイロ
-11	
-10	
-9	ロンドン
-8	パリ・ローマ
-7	アテネ・カイロ
-6	モスクワ
-5	
-4	カラチ
-3	
-2	バンコック
-1	北京
0	**東京（基準）**
1	シドニー
2	
3	
4	**日付変更線**
-19	ホノルル・タヒチ
-18	
-17	サンフランシスコ
-16	
-15	シカゴ・ダラス
-14	ニューヨーク
-13	

音楽のたのしみ（名演奏家MZ80K氏を訪ねて）



今回のお話はコンピュータに音楽を演奏させるTEMPO，MUSIC命令についてです。
TEMPO命令は文字通り速度を指定するものです。楽譜上の音符は、音の高さ（半音、オクターブも含む）と音の長さにしたがって、ストリングに直し、MUSIC命令で演奏します。
詳しい説明に入る前にTEMPO，MUSIC命令の構成の概略を示します。

テンポを指定する：	TEMPO	遅い方から早い方へ1から7までの数または変数
メロディを演奏させる：	MUSIC	音符の集まりを指定するストリング変数
音符の指定：	オクターブの指定 ／ ♯（嬰音） ／ 音名 ／ 音符の長さ	

TEMPO X （X＝1～7）：速度の指定

TEMPO X は速さをXで指定する命令です。TEMPO命令が無い場合はTEMPO4とみなしてMUSIC命令を実行します。

30 TEMPO 1 最も遅い（Lento，Adagio）

30 TEMPO 4 中位の速度（Moderato）TEMPO1の4倍の速さ

30 TEMPO 7 最も早い（Molto.Allegro，Presto）TEMPO1の7倍の速さ

＊なおCPU基板上の68kΩ のボリュームによって、TEMPOの設定も変えることができます。

「楽譜はストリングにします。」(同氏談)

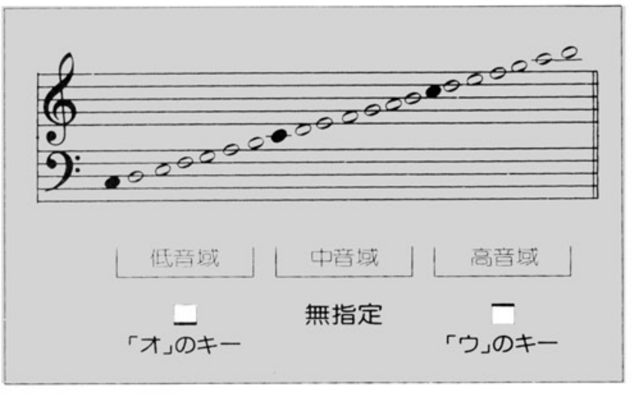
MUSIC"ストリング",M$(ストリング変数)

ストリングまたはストリング変数で指定される旋律や効果音を実際にスピーカから出す命令です。その速さは前にあるTEMPO文で指定された速さにしたがいます。

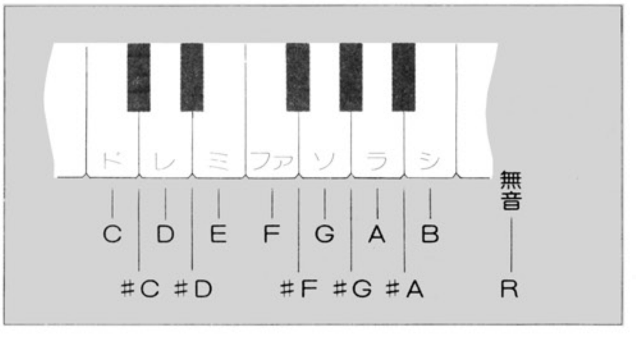
旋律や効果音をどのようにストリングに直すのかつぎに示します。
音符は音の高さ(オクターブと音名)、それに長さ(4分音符♩とか8分音符♪とか)によって指定されます。オクターブの指定、音名の指定、音長の指定の順に説明します。

オクターブの指定　⊔　⊓
コンピュータから出すことのできる音域は、右に示すように**3オクターブ**です。
たとえば黒点で示した音はCの音(ハ長調のド)ですが、3個はオクターブ指定によって区別されます。

　低音域のC………⊔C
　中音域のC………C
　高音域のC………⊓C



音名の指定　CDEFGAB　#F　R
音名の指定はCDEFGABと#記号を使います。
ドレミファ……とCDEF……の関係は右図に示してあります。半音の指定は**シャープ**記号#を使います。
休符(つまり無音)は**R**で指定します。



音長の指定
高さの指定された音の長さを指定します。32分音符から全音符までを**0から9**までの数字で指定します。(休符Rにつけると休符の長さになります)

32分休符	16分休符	付点16分休符	8分休符	付点8分休符	4分休符	付点4分休符	2分休符	付点2分休符	全休符
32分音符	16分音符	付点16分音符	8分音符	付点8分音符	4分音符	付点4分音符	2分音符	付点2分音符	全音符
0	1	2	3	4	5	6	7	8	9

同じ音長の音符が続くときは、2番目の音符からは音長の指定は省略できます。はじめから音長が指定されないと4分音符(5の音長)とみなして実行します。

音の大きさ
音の大きさをプログラムによって変えることはできませんが、音声ボリュームが本体の後部にあります。

プレリュード・アレグロ・アマービレ

楽譜をストリングに翻訳する練習をいくつかやってみましょう。前ページの表を参考にして下の楽譜を直してみます。
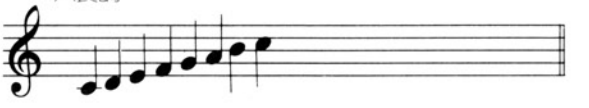
♪ はじめにハ長調、ニ長調、変ホ長調の音階。（４分音符で）

ハ長調　"CDEFGAB□C"
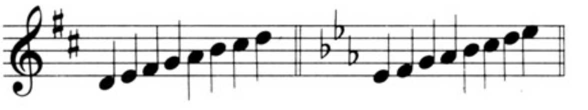
ニ長調　"DE＃FGAB□＃C□D"

変ホ長調"＃DFG＃G＃A□C□D□＃D"

ハ長調



ニ長調　　変ホ長調
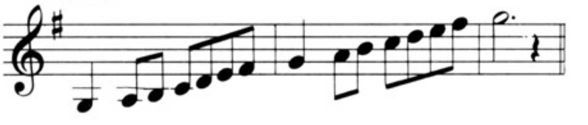


♪ ４分音符と８分音符でト長調の２オクターブの音階をG$に代入します。

G$＝"□G□A3□BCDE＃FG5
A3B□C□D□E□＃F□G8
R5"

（R5：４分休符）（□G8：付点２分音符の高いGの音）

ト長調
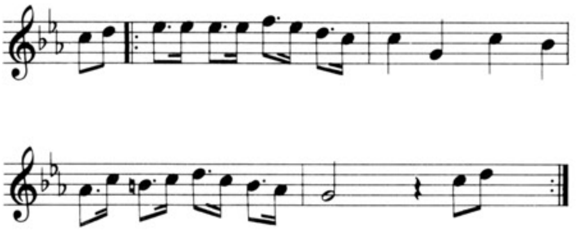




♪ ビートルズのGIRLの冒頭の部分をGR$に代入してみましょう。

GR$＝"□C3□D
□＃D4□＃D1□＃D4□＃D1
□F4□＃D1□D4□C1
□C5G□C＃A
＃G4□C1B4□C1
□D4□C1B4＃G1
G7R5"

GIRL（ジョンレノン、ポールマッカートニー）



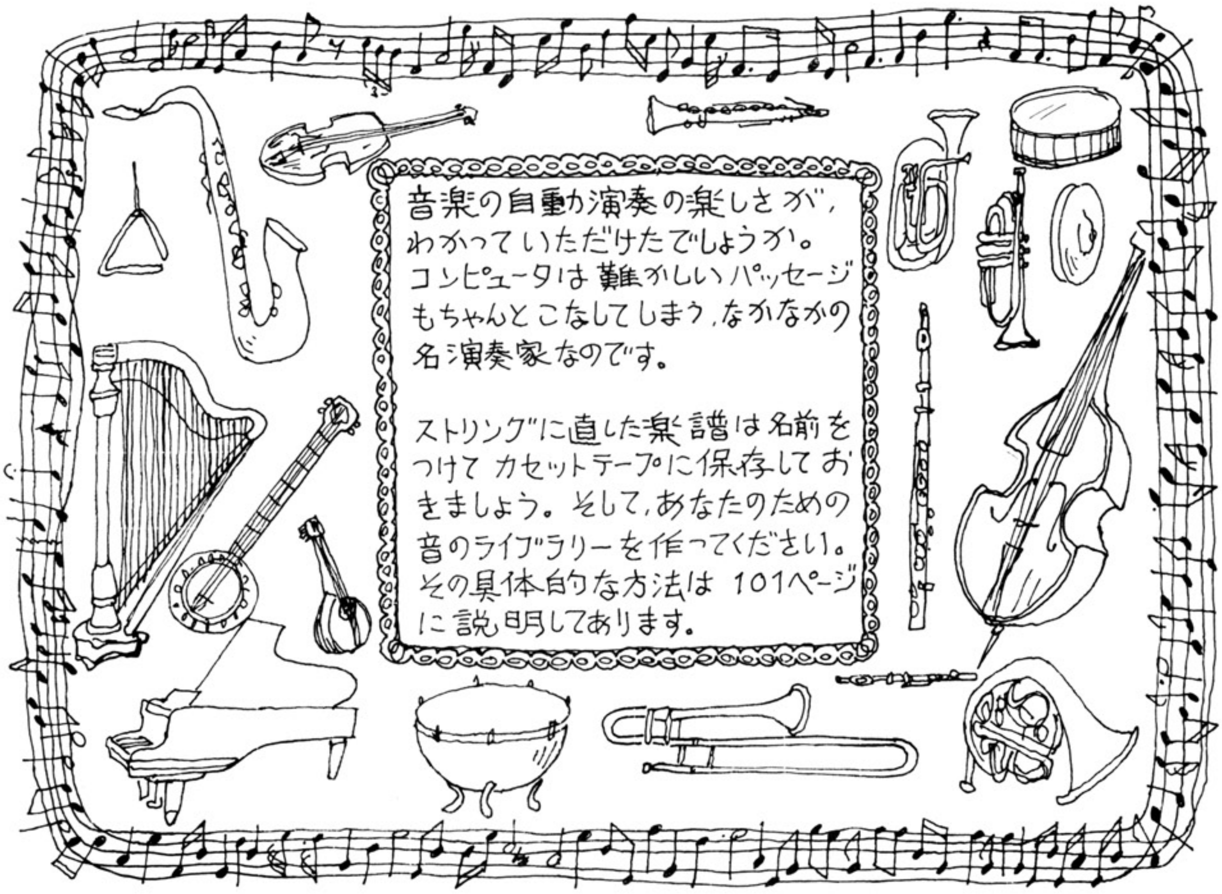
少し複雑そうに見えますが、慣れてしまうとわけのないことなのです。
長い曲の場合にはストリングの長さも長くなってきて、テレビ画面の２行内の命令文におさめることができなくなることがあります。この場合、曲をいくつかのストリング変数に入れておいて、ストリング変数の和をとってMUSIC命令のストリング変数とします。



あなたが作る音のライブラリー

♪簡単なオルゴールを作ってみます。 このオルゴールには二つの小さな曲が入っていて、1曲は明るい楽しい気分、 もう1曲は悲しい気分の曲です。 ストリングを適当に短かく区切って使っています。JかNをキー入力してください。

```
10  REM オルゴール (F. KROEPSCH ノ エチュード ヨリ)
20  J1$="C1□C□E□C□G□E□C□ECA□DA□F□DA□D"
30  J2$="GB□B□A□G□#F□F□DB□C□E□CBGFD"
40  J3$="C⊔BC⊔GECGE□C5R"
50  N1$="D⊔F⊔ADFA⊔G⊔#ADGA#A⊔A#CEG#AA
        GF□F□DAF"
60  N2$="DFA□D□F□AEG#A□E□G□#A□A□F□D□E
        □#CA□D#GAEF#C"
70  N3$="D⊔D⊔F⊔ADEF#CDFA□#C□D5R5"
100  INPUT"MAJOR OR MINOR? (J, N)";M$
110  IF M$="N" THEN 230
200  TEMPO 4
210  MUSIC J1$;J2$;J1$;J2$;J3$
220  GOTO 100
230  TEMPO 4
240  MUSIC N1$;N2$;N1$;N2$;N3$
250  GOTO 100
```



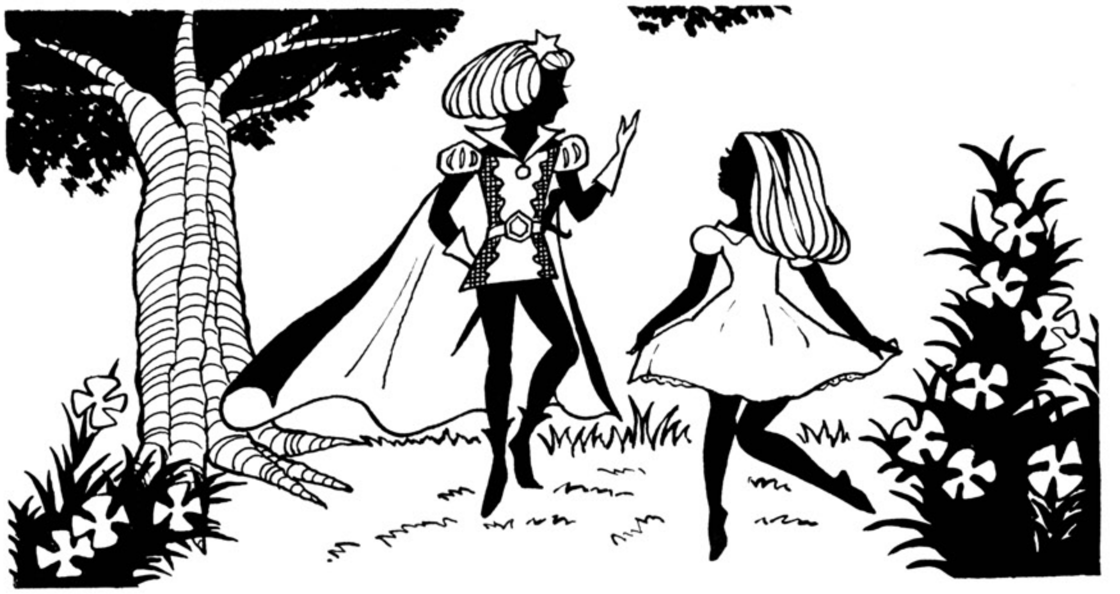
あすの朝7時にぜったい起きるんだ

エリスはねぼすけ。あしたの朝こそ早く起きて森の中をランニングしようと思っていても、起きると、もう学校へ行かなければなりません。
それをエリスの小さな星の王子様に話したら、こんなプログラムを教えてくれました。
………エリス、朝7時に起きるようにタイマーをセットするんだよ。エリスの好きなショパンが鳴ることだろう。そしたらぼくもそっと森へ降りてきみを待っているよ。



```
10 REM ショハ゜ン ノ マス゛ルカ
20 MM$="A3":M1$="A5□#C3□D□E□#F□G□#F0□G□#F4□E3□D□#CB"
30 M2$="A3□D2R0□D1□E2□D□#C3B□#C7□#C3"
40 M3$="A3□#C2R0□#C1□D2□#CB3A□D7□D3"
100 PRINT"C":INPUT"ケ゛ンサ゛イ ノ シ゛コクハ ?";TI$
110 INPUT"タイマー ノ セッテイ シ゛コクハ ?";TM$
120 PRINT"C"
130 PRINT"H↓↓↓↓↓";TAB(17);TI$
140 IF VAL(LEFT$(TI$,6))<>VAL(LEFT$(TM$,6)) THEN 130
150 TM$="9999"
160 TEMPO 3
170 MUSIC MM$,M1$,M2$,M1$,M3$,M1$,M2$,M1$,M3$
180 GOTO 120
```

最初のINPUT命令で現在時刻を入れます。そう、TI$に代入するわけです。6ケタの数、たとえば夜の8時ちょうどだったら200000として時報の瞬間に[CR]キーを押すんでしたね。そしてつぎにタイマーの時刻をキー入力します。たとえばエリスの場合は070000とすればいいのです。
少し手を加えて、ショパンのこのマズルカを全部入れておくのもいいでしょうし、時計もデザインしてみたらもっと楽しいタイマーができるはず。世界のどこにもないようなタイマーを持った時計をあなたの机の上に置くのも、あなたのプログラムの腕しだいです。



応用問題をふたつ

角度が０度から180度まで10度きざみに変わるときのSINの数表とグラフの波形です。

```
10  PRINT"C→";TAB(5);"SIN"
20  PRINT
30  FOR I=0 TO 180 STEP 10
40  X=I*π/180
50  S=SIN(X)
60  A=INT(10*S)
70  PRINT I;TAB(4);
80  PRINT S;TAB(18+A);
90  FOR J=0 TO A
100  PRINT"*";
110  NEXT J
120  PRINT
130  NEXT I
140  END
```

（40行）度単位の値をラジアン単位に変えています。



RUNさせてみてください。このグラフは少しラフですが、グラフの描き方については、後でいろいろ出てきますから、どうぞお楽しみに！

素数を求めるプログラムを作ってみましょう。素数とは、１以外の自分より小さいどんな整数でも割り切れない数のことです。最初の素数は２ですから、２の倍数、つまり２より大きい偶数はすべて素数ではありませんね。そこで添字つき変数を有効に使うために、はじめから偶数を除外して考えることにします。

```
10  DIM I(255)
20  FOR J=0 TO 255
30  I(J)=J*2+3:NEXT J
40  FOR K=0 TO SQR(255)
50  IF I(K)=0 THEN 90
60  KK= K+I(K)
70  FOR L=KK TO 255 STEP I(K)
80  I(L)=0:NEXT L
90  NEXT K
100  PRINT 2;
110  FOR M=0 TO 255
120  IF I(M)=0 THEN 140
130  PRINT I(M);
140  NEXT M
150  END
```

（10〜30行）添字つき変数Ｉ（　）の中に３から５１３までの２５６個の奇数を代入します。

（40〜90行）小さい方から素数を見つけ出し、その倍数の入っているＩ（　）の値を０にします。

（100〜150行）はじめに、唯一の偶数の素数「２」を表示し、次にＩ（０）の値が０でないもの、つまり素数を表示します。

少し複雑なプログラムですね。特に、素数の倍数を除外して行くところに注目してください。

名簿を作るひとにそっと教えたい……

よく名簿を作るときなど、名前を分類することがあります。こんなとき便利なプログラムを知っていると知らないのでは、天国と地獄の差はありますね。
ここでは、ストリングを分類する要領を勉強して、最高にステキな住所録や電話のメモ帳や家計簿を作るための足がかりにしましょう。

```
10  PRINT"ブンルイ スルノハ ナンニン デスカ"
20  INPUT X
30  DIM N$(X)
40  PRINT"ナマエ ヲ 1コズツ イレテクダサイ"
50  PRINT"タダシ 0 ナラ チュウシ シマス!"
60  FOR A=1 TO X
70  PRINT A;"ニンメ ヲ ドウゾ";
80  INPUT N$(A)                    ⇦ 氏名のキー入力
90  IF N$(A)="0" THEN 110
100 NEXT A
110 A=A-1
120 FOR B=1 TO A-1
130 FOR C=1 TO A-B
140 D=LEN(N$(C)):E=LEN(N$(C+1)):F=1:IF D<E THEN E=D
142 X=ASC(MID$(N$(C),F,1)):Y=ASC(MID$(N$(C+1),F,1))
    :IF X>Y THEN 150
144 IF X<Y THEN 180
145 IF (E=F)*(D=E) THEN 180
146 IF (E=F)*(D>E) THEN 150
148 F=F+1:GOTO 142
150 K$=N$(C)         ┐
160 N$(C)=N$(C+1)    ├ 前後を入れ替える
170 N$(C+1)=K$       ┘
180 NEXT C,B
190 PRINT
200 FOR B=1 TO A
210 PRINT N$(B)                    ⇦ 結果を表示する
220 NEXT B
230 END
```

キー入力した順の名前

TOM BROWN
HAROLD GREEN
JIM JONES
ANNE MILLER
TOM CARTER
ELICE THOMAS

⇨

分類した結果を見ると

ANNE MILLER
ELICE THOMAS
HAROLD GREEN
JIM JONES
TOM BROWN
TOM CARTER

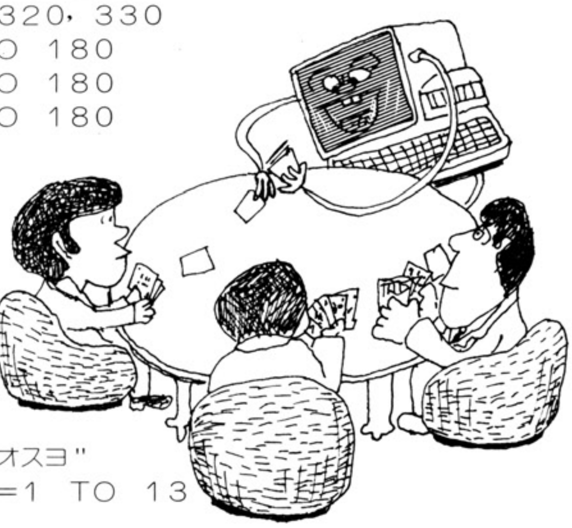




ポーカー野郎にカードを配らせると…

コンピュータがあなたにポーカーのカードを配ってくれます。とにかく、乱数を使ってマジメ(?)にカードをきってくれるから、イカサマの心配はありません。

```
10 DIM X(4,13)
20 C=0
30 PRINT:FOR A=1 TO 5
40 GOSUB 90:PRINT:NEXT A:PRINT
50 PRINT"アナタ ノ テ ハ コレデ イイカ"
60 INPUT"イイヨ(1) ツギ ヲ クレ(2) ? ";A
70 ON A GOTO 400,30
80 GOTO 50
90 C=C+1:IF C=51 THEN 500
100 M=INT(4*RND(1))+1
110 N=INT(13*RND(1))+1
120 IF X(M,N)=-1 THEN 100
130 X(M,N)=-1
140 IF N=1 THEN PRINT"ACE  :";:GOTO 180
150 IF N=10 THEN PRINT N;"  :";:GOTO 180
160 IF N<10 THEN PRINT N;"   :";:GOTO 180
170 ON N-10 GOTO 200,210,220
180 ON M GOTO 300,310,320,330
200 PRINT"JACK :";:GOTO 180
210 PRINT"QUEEN:";:GOTO 180
220 PRINT"KING :";:GOTO 180
300 A$="♠":GOTO 340
310 A$="♥":GOTO 340
320 A$="◆":GOTO 340
330 A$="♣"
340 FOR B=1 TO N
350 PRINT A$;
360 NEXT B
370 RETURN
400 PRINT
410 PRINT"ソレジャ マタ キリナオスヨ"
420 FOR M=1 TO 4:FOR N=1 TO 13
430 X(M,N)=0
440 NEXT N,M:GOTO 20
500 PRINT
510 PRINT"ノコリ ハ 2マイ ダヨ...マダ ヤルカイ !"
520 INPUT"ハイ(1),イイエ(2) ? ";B
530 ON B GOTO 400,550
540 GOTO 510
550 STOP
RUN
```



プログラムのポイントは…

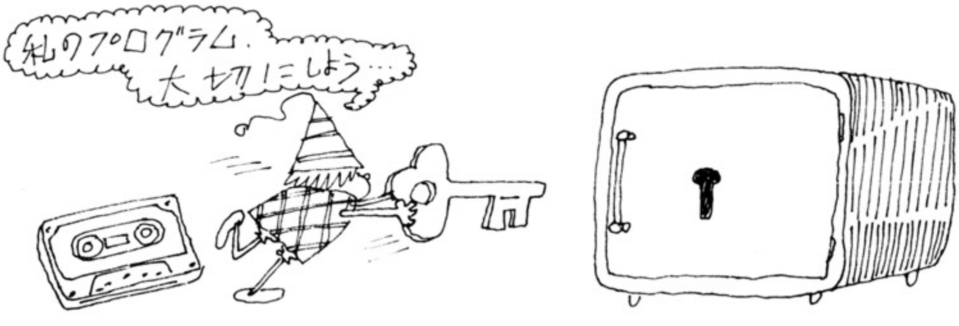
- 文番号100、110――新しくカードをめくっているところ
- 文番号120――新しくめくったカードが前にあったものなら、もういちどやり直す。
- 文番号130――配ってしまったカードに"−1"としるしをつけておく
- 文番号420〜440――全カードを集めて、カードのしるしを"0"に戻す。

プログラムの記録(SAVE)



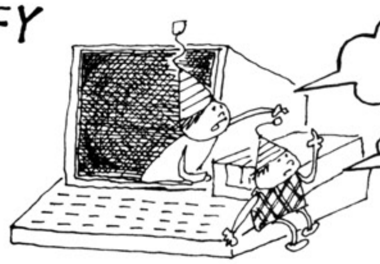


	SAVE [CR] のキー操作 / SAVE "NAME" [CR] のキー操作	NAMEの部分は16字以内のあなたのお好きな言葉
テレビ画面表示	⭳ RECORD・PLAY	記録準備OK!!
カセットテープレコーダの操作	[RECORD] [PLAY] ボタンを同時に押す	記録START!!
テレビ画面表示	WRITING NAME	ただいま記録中
	プログラムにファイル名がなければNAMEの表示はない.	
テレビ画面表示	READY	記録完了!!



VERIFYとLOADの使い方

VERIFY



(1) テープの巻きもどし → (2) **VERIFY "NAME" CR**
ファイル名がなければ "NAME" は不要

(3) ⯯ **PLAY** ……テレビ画面に表示 → (4) カセットテープレコーダの **PLAY** ボタンを押す．

(5) **FOUND ****** ……VERIFYする前に，他のプログラムがあるときに表示。ファイル名があればそれを****と表示．

(6) **FOUND NAME** ……VERIFYすべきプログラムを見つけたという表示．

(7) **VERIFYING NAME**
正確に記録しているかどうかをチェック中

(8) **DATA ERROR**
READY
記録に間違いあり，これが出たらもう一度**SAVE**しなおし

(8) **OK**
READY
この表示が出ればOK

LOAD



(1) **LOAD "NAME" CR**

(2) ⯯ **PLAY**

(3) **PLAY** ボタンを押す

(4) **FOUND ******

(5) **FOUND NAME**

（(2)〜(5)）**VERIFY**の(3)〜(6)と同じ表示および操作
ただし，(4)，(5)は**LOAD**に対する表示

(6) **LOADING NAME** ……コンピュータに読み出し中

(7) **READY** ……読み出し完了

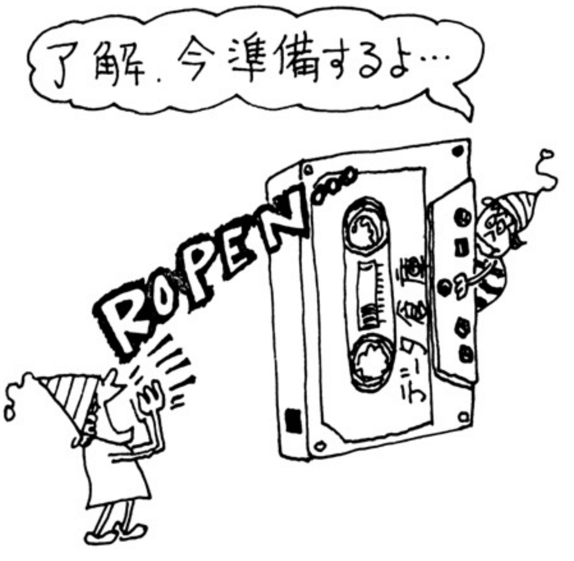
データだってカセットテープにとっておける

プログラムの保存ができるなら、データの保存もしたいもの……。
そんなひとはあと5個の命令を覚えるだけでカセットテープをデータの倉庫として自由自在にあやつれるようになりますよ。

- **WOPEN**………… データを書き込むための準備をします。ひとまとまりのデータに名前をつけるのもこの命令の役目です。
- **PRINT／T**…… PRINTと同じ使い方で、カセットテープにデータを書き込む命令です。
- **ROPEN**………… 反対に、データを読み出すための準備をします。名前のついたデータのかたまりを捜してくれる命令です。
- **INPUT／T**…… INPUTと同じ使い方でカセットテープからデータを読み出す命令です。
- **CLOSE**………… WOPENをしたらROPENする前に、またROPENをしたらWOPENする前に、この命令をかならず実行して下さい。

では、てはじめにデータを保存するため、1から99までの数字をカセットテープに書き込んでみます。ここで文番号10の"DATA"は、これから書き込むデータのかたまりにつけた名前ですから、16文字以内であれば、どんな内容でもかまいません。もちろん、名前をつけなくてもいいのです。

```
10  WOPEN  "DATA"
20  FOR  X=1  TO  99
30  PRINT/T  X
40  NEXT  X
50  CLOSE
60  END
```



どうです。うまく書き込みましたか？……それなら、こんどはいま書いたデータを読み出す番です。まず、カセットテープを巻き戻してから。

```
10  ROPEN  "DATA"
20  FOR  X=1  TO  99
30  INPUT/T  A
40  PRINT  A
50  NEXT  X
60  CLOSE
70  END
```

このとき文番号20で、99のかわりに100として、もういちどRUNしてみませんか？…… こんどはエラーになるはず。なぜなら100個目のデータは、もともと書き込んでいなかったからですね。でも、こんなことはよくあることなのです。書き込んだデータの数なんて、いちいち覚えてはいられません。そこで書き込みデータの終わりの印として、およそ関係のない数字、たとえば、−99999999などを書きそえておく方法がとられます。

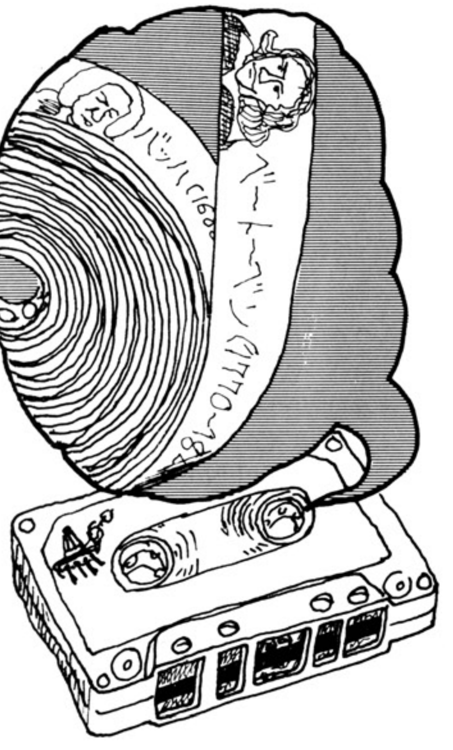
```
10  WOPEN  "DATA"
20  FOR  X=1  TO  99
30  PRINT/T  X
40  NEXT  X
50  PRINT/T  -99999999
60  CLOSE
70  END
```

```
10  ROPEN  "DATA"
20  FOR  X=1  TO  200
30  INPUT/T  A
40  IF  A=-99999999  THEN  70
50  PRINT  A
60  NEXT  X
70  CLOSE
80  END
```

音楽史の丸暗記術

データの保存、読み出し命令はストリング型のデータにももちろん使えます。
ここでは5人の作曲家の名前をカセットテープに書き込み、それを読み出すことをやってみました。

```
10 DIM N$(5)
20 N$(1)="BACH"
30 N$(2)="MOZART"
40 N$(3)="BEETHOVEN"
50 N$(4)="CHOPIN"
60 N$(5)="BRAHMS"
70 WOPEN "GREAT MUSICIANS"
80 FOR J=1 TO 5
90 PRINT/T N$(J)
100 NEXT J
110 CLOSE
120 END
```



数値データの書き込みと同じですね。そして読み出しは

```
200 DIM M$(5)
210 ROPEN "GREAT MUSICIANS"
220 FOR K=1 TO 5
230 INPUT/T M$(K)
240 PRINT M$(K)
250 NEXT K
260 CLOSE
```

これで完了です。ところで今の場合書き込みのとき使ったストリング変数N$( )と、読み出しのためのストリング変数M$( )は名前が異なっています。カセットテープにはデータとしての値そのものが収録されるので、それまで代入されていた変数の名前とは無関係になるからです。このため、読み出すデータがストリングならストリング変数で、また数値データなら数値変数で読みだす限り、プログラムで自在に変数名を変えてもかまわないわけです。
ここまでわかったところで、数値データとストリングデータの混じったデータファイルを作ってみましょう。たとえば、前の作曲家の没年を一緒に書き込んでおくには、前のプログラムにつぎの変更を加えた命令文を与えればいいのです。

```
15 DIM D(5),T(5)
65 D(1)=1750:D(2)=1791:D(3)=1827
67 D(4)=1849:D(5)=1897
90 PRINT/T N$(J),D(J)
```

これによって作成されるファイルが、ストリングデータと数値データが交互に並んだものになることはおわかりですね。ですからこのファイルの読み出しもそれに合わせなくてはなりません。文番号200、230、240は、こんなときには

```
200 DIM M$(5),T(5)
230 INPUT/T M$(K),T(K)
240 PRINT M$(K),T(K)
```

とされなくてはなりません。もとのままだとストリング変数M$( )へ数値データが入ろうとしてエラーを生じることになります。

イキな先生がつくった成績リスト

あるクラスの国語、英語、理科、数学の成績をカセットテープに記録しておくためのプログラムです。
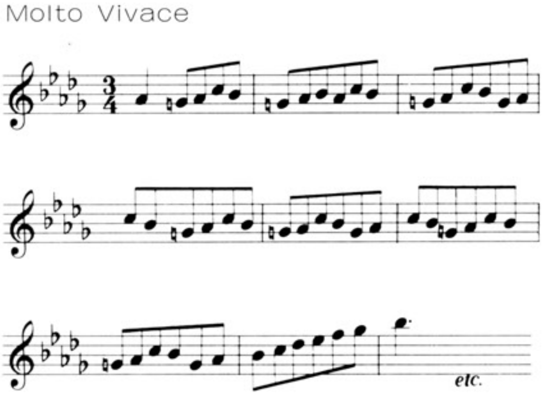


```
10  INPUT"ﾅﾝﾆﾝ ﾃﾞｽｶ ?";N
20  DIM N$(N),K(N),E(N)
30  DIM R(N),S(N)
40  A$=" ﾉ ﾃﾝｽｳ ﾊ "
50  FOR X=1 TO N
60  PRINT:PRINT X;"ﾆﾝﾒ"
70  INPUT"ﾅﾏｴ ｦ ﾄﾞｳｿﾞ? ";N$(X)
80  PRINT"ｺｸｺﾞ";A$;:INPUT K(X)
90  PRINT"ｴｲｺﾞ";A$;:INPUT E(X)
100 PRINT"ﾘｶ";A$;:INPUT R(X)
110 PRINT"ｽｳｶﾞｸ";A$;:INPUT S(X)
120 NEXT X
130 WOPEN"ｾｲｾｷ"          ←「セイセキ」という名前のデータ群を次から書くため
140 PRINT/T N            ←クラスの人数の書きこみ
150 FOR X=1 TO N
160 PRINT/T N$(X),K(X),E(X),R(X),S(X)  ←点数の書き込み
170 NEXT X
180 CLOSE                ←書き込みの終了
190 END
```

それでは書き込んだ成績データを読み出して、各人の平均点と各教科の平均点を計算してみましょう。

```
10  ROPEN"ｾｲｾｷ"                 ←「セイセキ」という名前のデータ群を捜すため
20  INPUT/T N                   ←クラスの人数の読み出し
30  DIM N$(N),K(N),E(N)
40  DIM R(N),S(N)
50  FOR X=1 TO N
60  INPUT/T N$(X),K(X)          ←名前と国語の点数の読み出し
70  INPUT/T E(X),R(X),S(X)      ←英語、理科、数学の点数の読み出し
80  NEXT X
90  CLOSE                       ←読み出しの終了
100 PRINT TAB(10);"ｺｸｺﾞ";
110 PRINT TAB(15);"ｴｲｺﾞ";
120 PRINT TAB(20);"ﾘｶ";
130 PRINT TAB(25);"ｽｳｶﾞｸ"
140 FOR X=1 TO N
150 PRINT N$(X);TAB(10);K(X);
160 PRINT TAB(15);E(X);
170 PRINT TAB(20);R(X);
180 PRINT TAB(25);S(X);
190 PRINT TAB(30);(K(X)+E(X)+R(X)+S(X))/4
200 K(0)=K(0)+K(X):E(0)=E(0)+E(X)
210 R(0)=R(0)+R(X):S(0)=S(0)+S(X)
220 NEXT X
230 PRINT TAB(10);K(0)/N;TAB(15);E(0)/N;
240 PRINT TAB(20);R(0)/N;TAB(25);S(0)/N
250 END
```

音楽のライブラリーをテープにつくる

ＭＵＳＩＣ命令の項でお話しした「音楽のライブラリー」を作るのにこのデータファイルは欠かせないものです。
楽曲のデータとなるものは具体的にはいろいろな記号の集まりからなるストリングデータでした。楽曲ごとにこのデータの集まりに名前を付けておけば、カセットテープに記録されたたくさんの曲の中から、名前を指定した曲だけピックアップすることができることになります。



ショパン作曲「小犬のワルツ」

たとえば、タイマーのオルゴールに使う曲をこのライブラリーの中からとり出して変更したり、ゲームのプログラムやグラフィックプログラムに採用したり、自由自在に活用できることうけあい！……

91ページで使ったクレプシュの練習曲をデータファイルに書き込むには

```
300  WOPEN  "レンシュウキョク"
310  PRINT/T  J1$, J2$, J3$, N1$, N2$, N3$
320  CLOSE
```

とすればいいのです。ここで注意しなくてはならないのは書き込みのデータの文字数は２５５文字以下でなくてはならないことです。したがって

```
305  MA$=J1$+J2$+J3$:MB$=N1$+N2$+N3$
310  PRINT/T  MA$, MB$
```

などとすると、ストリング変数ＭＡ＄，ＭＢ＄の内容は２５５文字を越えてしまい、データの書き込みは不完全なものとなってしまいます。

曲の長さは曲によって異なるように、ストリングデータの長さも曲によってまちまちなはず。そうしたデータを読み出す際にデータエラーを起こさないように、一曲一曲終曲を示すデータを最後に書いておくのです。たとえば、終曲記号を"◧■"と決めておけば、どの一曲も１００個以内のストリングデータから成っているとして

```
500  ROPEN  "コイヌ ノ ワルツ"
510  FOR  A=1  TO  100
520  INPUT/T  M$(A)
530  IF  M$(A)="◧■"THEN  550
540  NEXT  A
550  CLOSE
```

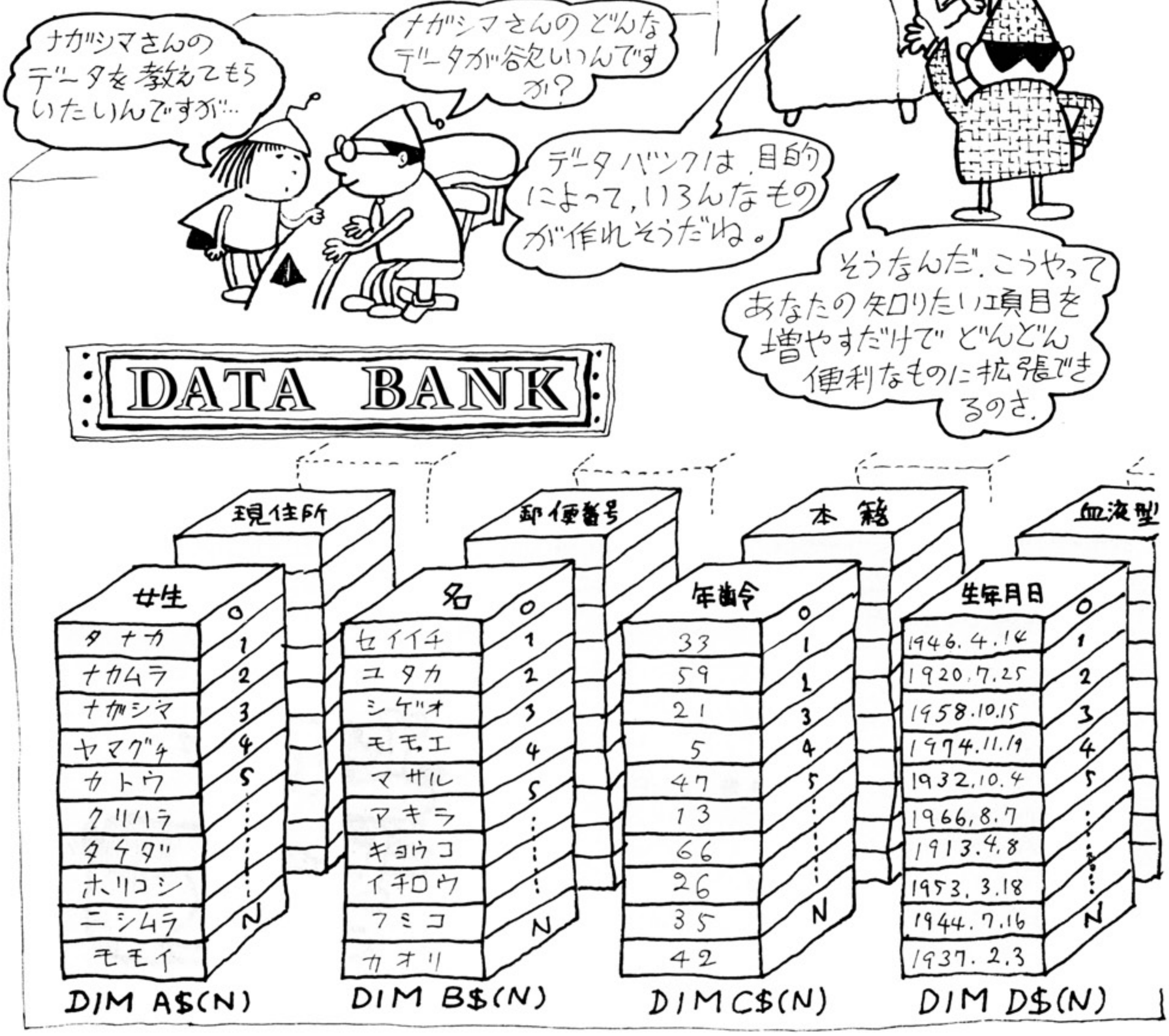


とすれば、「小犬のワルツ」がきちんと読み出されます。

データバンクはコンピュータのお家芸

コンピュータの記憶力はばつぐんなので、いろんな項目をふくんだ名簿をつくるぐらい訳ありません。項目の順序を決めておいて、たとえば、姓名—年令—生年月日—現住所—本籍地という具合にデータを記憶させておくのです。そしてデータがちょうど銀行にたくわえられるお金のようにたくさん集められると、データバンク、つまりデータの銀行と呼ばれるようになるわけです。
ただ、ほんとうの銀行とちがって、データバンクになると検索や分類がプログラム次第ですぐにできることでしょう。
検索というのは、知りたいと思う項目を指定して、それに応じたすべてのデータを捜しだすことです。たとえば血液型を指定して、同じ血液型のひとをすべて捜しだすようなとき、血液型によって「検索する」といいます。
また、ある項目に注目して、たとえば名前のアイウエオ順に並びかえるとか、年令の小さい順に並びかえたりすることを「分類する」といいます。





電話番号簿もデータバンクのひとつ

以上の内容を理解したところで、DATA命令にストリングとして各データが記憶されている場合のプログラムをまとめてみました。もちろん、このプログラム例をもとに、郵便番号や住所を同時に知ることのできるような変更も楽しい勉強になることでしょう。憶することなく挑戦してみたら、きっと便利なデータバンクになると思います。



```
10 N=12
20 DIM M$(N)  ← 姓
30 DIM N$(N)  ← 名
40 DIM A$(N)  ← 自宅の電話の市外局番
50 DIM B$(N)  ← 自宅の電話の市内局番
60 DIM C$(N)  ← 勤務先電話の市外局番
70 DIM D$(N)  ← 勤務先電話の市内局番
80 DIM F(N)
90 FOR K=1 TO N
100 READ M$(K),N$(K)
110 READ A$(K),B$(K)  ⇦ 各データの読み込み
120 READ C$(K),D$(K)
130 NEXT K
140 PRINT:PRINT:X=0
150 PRINT"ナマエ ハ ナンテ゛スカ";
160 INPUT X$              ← 検索したい人の姓をキー入力する
170 FOR K=1 TO N
180 IF M$(K)=X$ THEN X=X+1:F(X)=K  ← 姓を使った検索をする
190 NEXT K
200 IF X<>0 THEN 240
210 PRINT"カ゛イトウシャ カ゛ イマセン!"
220 PRINT"モウイチト゛ ト゛ウソ゛"
230 GOTO 140
240 PRINT:PRINT
250 FOR K=1 TO X
260 L=F(K)                ← 同じ姓の人でも表示するためのものです。
270 PRINT"セイメイ ハ ";M$(L);" ";N$(L)
280 PRINT"シ゛タク ノ テ゛ンワ: ";
290 PRINT"(";A$(L);") ";B$(L)
300 PRINT"キンムサキ テ゛ンワ : ";
310 PRINT"(";C$(L);") ";D$(L)
320 NEXT K
330 GOTO 140
340 DATA カワク゛チ,オサム,0745,33-1017,06,881-2063
350 DATA ナカニシ,カオル,03,901-8366,045,533-1185
360 DATA オカ゛タ,ケンシ゛,0422,65-8111,03,761-0745
370 DATA ミス゛タニ,タケコ,045,490-2951,045,637-5300
380 DATA タニムラ,マスシ゛,03,294-4045,0472,44-2251
390 DATA ワタ゛,コウシ゛,03,783-2544,0429,32-3024
400 DATA マエタ゛,ノリオ,0425,28-4681,06,961-5162
410 DATA カンチ,マサヒロ,0582,32-4695,066,341-6839
420 DATA ヤマタ゛,ヒロユキ,044,776-8101,044,604-7705
430 DATA オオウチ,リュウイチ,06,812-9230,06,621-1221
440 DATA サイトウ,ミツオ,043,567-8901,043,231-4567
450 DATA ツシ゛タ,マサヒロ,06,901-1023,06,591-5221
460 END
```

モールス信号でSOS SOS

モールス信号は1838年アメリカの画家、サミュエル・モースが考えだしたもので、いまでも大切な通信手段のひとつです。
原理は簡単です。ある一定の周波数の電波を出す時間と出さない時間の比率を定めているわけです。

長音（ツー）━━（短音の3倍の時間の送信）

短音（トン）━

休止　　　　　　（短音と同じ時間の無音）

この3つの組み合わせで必要な符号を表現する約束がモールスコード表です。その一部分を下のほうに示していますので、机の上でたとえばSOSと打ってみてください。
あんがいむずかしいって？……コンピュータのキー入力だって、使っているうちに自然に指が動くようになったはず。モールス信号もまず頭ではなく、自然に指が動くようになるまで練習が肝心ですね。
そこでつぎのページのプログラムはモールス信号を作りだして、簡単に練習できるようにしたものです。文番号20から文番号270までがAからZまで、文番号290から文番号380までが0から9までの信号をつくりだすためのストリングです。簡単に説明してみましょう。



□A5――→高音域のA（ラ）の音で音長は5（モールス信号のツーに当たります）

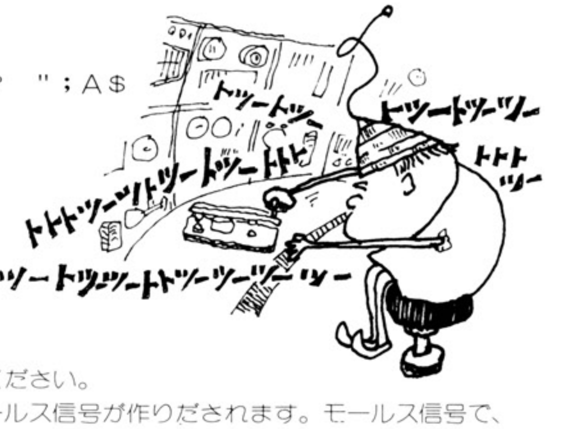
□A2――→高音域のA（ラ）の音で音長は2（モールス信号のトンに当たります）

R2――→音のない部分でその長さが2

イA	・－	ヲJ	・－－－	ムT	－	エ	－・－－－	モ	－・－・	8	－－－・・
ロ	・－・－	ワK	－・－	ウU	・・－	テ	・－・－－	セ	・－－－・	9	－－－－・
ハB	－・・・	カL	・－・・	ヰ	・－・・－	ア	－－・－－	ス	－－－・－	0	－－－－－
ニC	－・－・	ヨM	－－	ノ	・・－－	サ	－・－・－	ン	・－・－・	カンマ	・－・－・－
ホD	－・・	タN	－・	オ	・－・・・	キ	－・－・・	1	・－－－－	ダンラク	・－・－・・
ヘE	・	レO	－－－	クV	・・・－	ユ	－・－－	2	・・－－－	かっこ	－・－－・－
ト	・・－・・	ソ	－－－・	ヤW	・－－	メ	－・・・－	3	・・・－－	とじかっこ	・－・・－・
チF	・・－・	ツP	・－－・	マX	－・・－	ミ	・・－・－	4	・・・・－	終信	・・・－・
リG	－－・	ネQ	－－・－	ケY	－・－－	シ	－－・－・	5	・・・・・	ダク点（゛）	・・
ヌH	・・・・	ナR	・－・	フZ	－－・・	ヱ	・－－・・	6	－・・・・	半ダク点（゜）	・・－－・
ル	－・－－・	ラS	・・・	コ	－－－－	ヒ	－－・・－	7	－－・・・	長音（ー）	・－－・－

ツーツートントンツートントン

```
10 DIM A1(100),M$(127)
20 M$(65)="□A2R2□A5"
30 M$(66)="□A5R2□A2R2□A2R2□A2"
40 M$(67)="□A5R2□A2R2□A5R2□A2"
50 M$(68)="□A5R2□A2R2□A2"
60 M$(69)="□A2"
70 M$(70)="□A2R2□A2R2□A5R2□A2"
80 M$(71)="□A5R2□A5R2□A2"
90 M$(72)="□A2R2□A2R2□A2R2□A2"
100 M$(73)="□A2R2□A2"
110 M$(74)="□A2R2□A5R2□A5R2□A5"
120 M$(75)="□A5R2□A2R2□A5"
130 M$(76)="□A2R2□A5R2□A2R2□A2"
140 M$(77)="□A5R2□A5"
150 M$(78)="□A5R2□A2"
160 M$(79)="□A5R2□A5R2□A5"
170 M$(80)="□A2R2□A5R2□A5R2□A2"
180 M$(81)="□A5R2□A5R2□A2R2□A5"
190 M$(82)="□A2R2□A5R2□A2"
200 M$(83)="□A2R2□A2R2□A2"
210 M$(84)="□A5"
220 M$(85)="□A2R2□A2R2□A5"
230 M$(86)="□A2R2□A2R2□A2R2□A5"
240 M$(87)="□A2R2□A5R2□A5"
250 M$(88)="□A5R2□A2R2□A2R2□A5"
260 M$(89)="□A5R2□A2R2□A5R2□A5"
270 M$(90)="□A5R2□A5R2□A2R2□A2"
280 REM NO.
290 M$(48)="□A5R2□A5R2□A5R2□A5R2□A5"
300 M$(49)="□A2R2□A5R2□A5R2□A5R2□A5"
310 M$(50)="□A2R2□A2R2□A5R2□A5R2□A5"
320 M$(51)="□A2R2□A2R2□A2R2□A5R2□A5"
330 M$(52)="□A2R2□A2R2□A2R2□A2R2□A5"
340 M$(53)="□A2R2□A2R2□A2R2□A2R2□A2"
350 M$(54)="□A5R2□A2R2□A2R2□A2R2□A2"
360 M$(55)="□A5R2□A5R2□A2R2□A2R2□A2"
370 M$(56)="□A5R2□A5R2□A5R2□A2R2□A2"
380 M$(57)="□A5R2□A5R2□A5R2□A5R2□A2"
390 REM"SPACE"
400 M$(32)="R5"
1000 INPUT "ｺｰﾄﾞﾊ ﾄﾞｳｼﾏｽ ｶ ? ";A$
1010 FOR J=1 TO LEN(A$)
1020 A1(J)=ASC(MID$(A$,J,1))
1030 NEXT J
1040 FOR J=1 TO LEN(A$)
1050 MUSIC M$(A1(J)),"R5"
1060 NEXT J
1070 GOTO 1000
```



あなたはアルファベットのA～Z、数字の0～9をキー入力してください。
例えば、I LOVE YOUとキー入力すればそれに応じたモールス信号が作りだされます。モールス信号で、あなたの彼女に愛の告白を!!

そして果てしなくつづく「時」…

BASIC入門の終わりに「万年暦」のプログラムを置くことにしました。詳しい説明は要らないと思います。私たちの「時」は果てしなく続いて行きます。

```
5 DIM M$(12),W$(7)
10 FOR I=1 TO 12:READ M$(I):NEXT I
20 FOR I=1 TO 7:READ W$(I):NEXT I
30 INPUT"年 ヲ ﾄﾞｳｿﾞ? ";Y:INPUT"月 ヲ ﾄﾞｳｿﾞ? ";MT
40 H=MT:GOSUB 400:K2=YB+1
50 H=MT+1:GOSUB 400:K1=YB+1
60 I=K1-K2:IF I>=0 THEN L=28+I:GOTO 70
65 L=35+I
70 IF MT=12 THEN L=31
75 PRINT"🅲":GOSUB 190
80 PRINT TAB(8);Y;"  ";M$(MT):T=4
90 FOR I=1 TO 7:PRINT TAB(T);W$(I);:T=T+4
   :NEXT I :PRINT
100 T=0:IF K2=0 THEN 120
110 FOR I=1 TO K2:PRINT TAB(T);:T=T+4:NEXTI:T=T-4
120 FOR I=1 TO L:I$=STR$(I):J=LEN(I$)
130 PRINT TAB(T+5-J);I$;:T=T+4
140 IF T=28 THEN T=0:PRINT
150 NEXT I
160 IF T<>0 THEN PRINT
170 GOSUB 190
180 PRINT"⇩":GOTO 30
190 FOR Z=1 TO 31:PRINT"*";:NEXT:PRINT:RETURN
200 DATA JAN,FEB,MAR,APR,MAY,JUN
210 DATA JUL,AUG,SEP,OCT,NOV,DEC
220 DATA 日,月,火,水,木,金,土
230 END
400 X=Y
410 I=H-3:J=12:GOSUB 600:MM=Z
420 IF MM>9 THEN X=X-1
430 I=X:J=400:GOSUB 600:X=Z
440 X4=INT(X/4):X1=INT(X/100)
450 KY=X+X4-X1
460 I=MM:J=5:GOSUB 600:MZ=Z
470 M5=INT(MM/5):M2=INT(MZ/2)
480 I=MZ:J=2:GOSUB 600:P=Z
490 KM=13*M5+5*M2+3*P
500 I=KY+KM+3:J=7:GOSUB 600:YB=Z
510 RETURN
600 REM Z=I,J
610 K=INT(I/J)
620 Z=I-K*J
630 IF Z<0 THEN Z=Z+J
640 RETURN
```

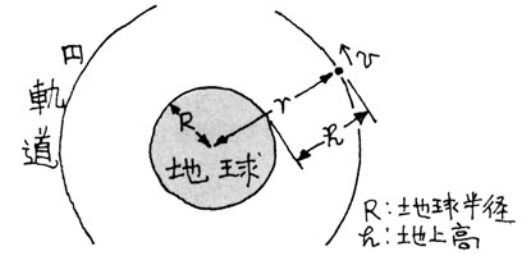




宇宙豆事典

あなたが、天文学とか人工衛星など、宇宙に興味がおありでしたら、コンピュータを使って、いろいろと計算をしたり図表を作ったりしてみてはいかがでしょう。ここでは、そのために必要な代表的な計算式や数値を示します。

宇宙での物体の動きは、地上とは違って空気抵抗などによる複雑さもなく、数学的計算とよく一致するはずです。ただし、より正確な値となると、他の天体による影響や、地球の形状のひずみによる摂動、希薄とはいえ宇宙に存在するガス圧まで考慮に入れたものでなくてはなりません。たとえば、高度 800kmの宇宙では10^{-9}mmHgの空気があるし、約36,000kmの高度にある静止衛星は、他の天体の影響により軌道傾斜角が1年に約1度変わってしまいます。



円軌道を描いて運動する衛星の速度： $v=\sqrt{\dfrac{G \cdot M}{r}}$ 〔m／s〕

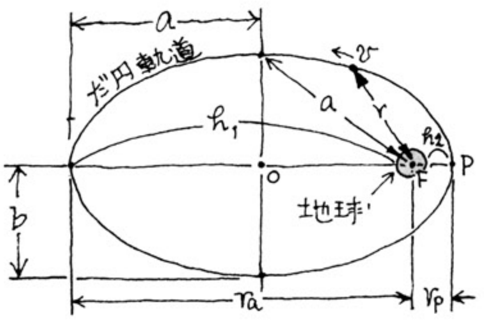
周期： $T=2\pi\sqrt{\dfrac{r^3}{G \cdot M}}$ 〔s〕

$G=6.67\times10^{-11}$ 〔N・m^2／kg^2〕
万有引力定数

M：地球の質量 〔kg〕

r：地球の中心から衛星までの距離〔m〕

だ円軌道を描いて運動する衛星の速度： $v=\sqrt{G \cdot M\left(\dfrac{2}{r}-\dfrac{1}{a}\right)}$ 〔m／s〕



周期： $T=2\pi\sqrt{\dfrac{a^3}{G \cdot M}}$ 〔s〕

長半径： $a=\dfrac{h_1+h_2+2R}{2}$ 〔m〕

短半径： $b=\sqrt{a^2-(a \cdot e)^2}$ 〔m〕

離心率： $e=OF/OP=\dfrac{\sqrt{a^2-b^2}}{a}$

$r_p=a(1-e)$ 〔m〕

$r_a=a(1+e)$ 〔m〕

	質量(太陽を1とする)	赤道半径	離心率	太陽からの平均距離(a)
太陽	1.000	69 6000km	−	−
水星	0.166×10^{-6}	2440	0.20563	0.57910×10^8km
金星	2.448×10^{-6}	6056	0.00678	1.08210 〃
地球	30.034×10^{-7}	6378	0.01672	1.49600 〃
火星	3.227×10^{-7}	3390	0.09338	2.27944 〃
木星	95.479×10^{-5}	7 1400	0.04829	7.7834 〃
土星	2.856×10^{-4}	6 0400	0.05604	14.2700 〃
天王星	4.373×10^{-5}	2 3700	0.04613	28.7103 〃
海王星	5.178×10^{-5}	2 5110	0.01004	44.971 〃
冥王星	0.552×10^{-6}	3400	0.24842	59.136 〃
月	3.694×10^{-8}	1738	0.0549*	38 4400km*

太陽の質量＝1.991×10^{30}kg　　＊地球に対する値　　理科年表より
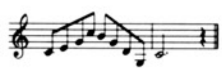
BASIC命令のまとめ SP-5030

直接実行命令

命令	書式	説明
LOAD	LOAD	カセットテープファイル中のBASICプログラムを読み込みます。この場合、最初に見つかったプログラムを読み込むことになります。
	LOAD "NAME" (←ファイル名)	"NAME"というファイル名のついたBASICプログラムを読み込みます。
	LOAD "NAME OBJ" (←機械語ファイル名)	読み込むべきファイルが機械語ファイルであった場合、そのローディングアドレスが、LIMIT命令で限定したBASICエリア以降の機械語プログラムエリア内であれば、そこへローディングを行います。(このときBASICテキストエリアはNEWされません。)
SAVE	SAVE	現在、コンピュータ内にあるプログラムをカセットテープに記憶させます。
	SAVE "NAME" (←ファイル名)	上記と同様にプログラムを"NAME"というファイル名をつけて記憶させます。
VERIFY	VERIFY	現在、コンピュータ内にあるプログラムとカセットテープファイル中のBASICプログラムと比較します。
	VERIFY "NAME" (←ファイル名)	上記と同様に、コンピュータ内のプログラムと、"NAME"というファイル名のついたプログラムを比較します。
CLR	CLR 10 CLR	数値をとる変数はすべて0、ストリング変数もすべて空にします。
RUN	RUN	プログラムを実行します。
	RUN 100	プログラムを文番号100から実行します。(RUNではその直後、すべての変数を0または空にします)
LIST	LIST	現在、コンピュータにあるプログラムのすべてをテレビ画面に表示します。
	LIST 70	文番号70のプログラムだけを表示します。
	LIST 70−	文番号70以降のプログラムをすべて表示します。
	LIST 70−100	文番号70から文番号100までのプログラムを表示します。
	LIST −100	文番号100までのプログラムを表示します。
LIST/P	LIST/P	上記のLISTとほとんど同じ機能をプリンターに対して行ないます。
NEW	NEW	現在、コンピュータ内にあるプログラムを消去します。
CONT	CONT	プログラムの実行がストップ(BREAKキーによって、あるいはプログラム中のSTOP命令、END命令によって)された際には、そのストップした所から実行を再開することができます。(ストップしたあとで、文番号を使ったプログラムの編集をしたら、CONTは使うことができません。直接実行命令は使ってもかまいません)
BYE	BYE	BASICの使用をやめて、モニターに戻ります。

ステートメント

GOTO	100 GOTO 200	文番号200へジャンプします。
GOSUB	100 GOSUB 700	文番号700のサブルーチンへ分岐します。
RETURN	800 RETURN	サブルーチンの実行を終了し、メインプログラムでGOSUB命令をした次のステートメントへ戻ります。
ON…GOTO	10 ON A GOTO 70,80,90	変数Aの値が1なら文番号70へ、2なら文番号80へ、3なら文番号90へジャンプします。Aが0または4以上なら次の文を実行します。ONにはINTが含まれておりAが2.7ならば2の場合と同様に文番号80へジャンプします。
ON…GOSUB	100 ON A GOSUB 700,800	変数Aの値が1なら文番号700、2なら文番号800のサブルーチンへ分岐します。Aが0または3以上なら次の文を実行します。
IF…THEN	10 IF A>20 THEN 100	変数Aの値が20より大なら、文番号100へジャンプします。Aが20以下なら次の文を実行します。
	30 IF B<3 THEN B=B+3	変数Bの値が3より小なら変数BにB+3の値を代入します。Bが3以上なら次の文を実行します。
IF…GOTO	100 IF A>=B GOTO 10	変数Aの値が変数Bの値以上なら文番号10へジャンプします。AがBより小なら次の文を実行します。
IF…GOSUB	100 IF A=B*2 GOSUB 700	変数Aの値が変数Bの値の2倍に等しいなら文番号700のサブルーチンへ分岐します。等しくないなら次の文を実行します。 （条件文のあとにマルチステートメントが来る場合、ON文は条件が成り立たないとき実行されますが、IF文は条件が成り立たないとき次の文番号へ移りマルチステートメントは無視されます。）
FOR…NEXT STEP	10 FOR A=1 TO 10 20 PRINT A 30 NEXT A	文番号10は変数Aを1から10まで変化させよという命令で、最初Aの値は1となります。文番号20はAの値をテレビ画面に表示させる命令ですから数値1が表示されます。次に文番号30でAの値は2になってこのループを繰り返します。こうしてAの値が10になるまでこのループが繰り返されます。(ループを終了した時点でAには11の値が入っています。)
	10 FOR B=2 TO 8 STEP 3 20 PRINT B↑2 30 NEXT	文番号10は変数Bを2から8まで、3ずつ大きくして変化させよという命令です。STEPの値を負にして変数の値を小さくして行くこともできます。
	10 FOR A=1 TO 3 20 FOR B=10 TO 30 30 PRINT A,B 40 NEXT B 50 NEXT A （20〜40：Bループ、10〜50：Aループ）	変数AとBについてFOR……NEXTループを重ねた例です。BループはAループの内部に置かれているところに注目して下さい。ループは2重3重……と重ねることができますが、内側のループは必ず外側のループ内に閉じていなくてはなりません。
READ…DATA		DATA文に置かれた定数またはストリングをREAD文の中に示された変数に代入する命令です。READ文中の変数と、それに対応するDATA文中の値は、数値変数なら数値、ストリング変数

		ならストリングと、変数の形が一致しなくてはなりません。
	10 READ H$,H,S$,S,D$,D 20 DATA ﾊｰﾄ,3 30 DATA ｽﾍﾟｰﾄﾞ,10 40 DATA ﾀﾞｲﾔ,6	左の例では、READ文の最初の変数、すなわちストリング変数H$にDATA文の最初のデータ、すなわちストリング"ハート"が代入されます。次に2番目の変数Hには、数値3が代入され、次々に読み込まれて行きます。
RESTORE		READ……DATA命令では、READ命令につれてDATA文中から読み込むデータが移って行きますが、RESTORE文を使うことによって読み込むデータをDATA文の最初に戻すことができます。
	10 READ A,B,C 20 RESTORE 30 READ D,E 40 DATA 3,6,9,12,15	左の例では、文番号10のREAD文によって、変数A、B、Cにそれぞれ、3、6、9の値が代入されますが、次にRESTORE文が置かれているので、文番号30のREAD文によって変数D、Eに代入される値は、12、15とはならずに、それぞれ3、6が代入されることになります。
MUSIC TEMPO		音楽の自動演奏をさせる命令です。文字通りTEMPO文で指定されたテンポにのって、MUSIC文にあるクォーテーション「"」内の旋律のストリング（音程と音長の指定された音符の集まりに相当します）を音にしてスピーカを鳴らします。
	300 TEMPO 7 310 MUSIC "DE#FGA"	文番号300でテンポが7（最も速い速度です）に指定されます。文番号310で、7のテンポで中音域のレミファの#ソラが4分音符の長さで続けて演奏されます。TEMPO文が無かったらテンポは4で演奏されます。
	300 M1$="C3EG□C" 310 M2$="B3GD□G" 320 M3$="C8R5" 330 MUSIC M1$,M2$,M3$	左の例では、旋律を3つのストリング変数に代入しておいてMUSIC命令をさせるものです。五線符で示すと右の旋律が演奏されます。なお、TEMPO文がありませんので、4のテンポが設定されています。 
DEF FN	100 DEF FNA(X)=X↑2−X 110 DEF FNB(X)=LOG(X) 120 DEF FNZ(Y)=LN(Y)	DEF FNで関数の定義をします。文番号100はX^2-XをFNA(X)に、文110は$\log_{10}X$をFNB(X)に、文120は$\log_e Y$をFNZ(Y)に定義します。関数は1変数に限ります。
SET RESET	SET X,Y RESET X,Y	SETの変数X、Yで指定されるテレビ画面上の座標を点灯させ、RESETで指定された点は消されます。X座標は画面の左から右へ0から79、Y座標は上から下へ0から49となっています。したがって画面の左上隅の座標は0、0であり、右下隅は79、49です。 X、Yの指定は定数以外に、変数、数式でもできます。
	10 FOR I=0 TO 49 20 SET I,I 30 NEXT	左のプログラムは、左上隅から画面の下端まで、斜線を描くものです。
	100 FOR I=0 TO 79 110 SET I,SQR(I) 120 NEXT	文番号100から120で、画面に放物線が描かれます。SQR（I）は整数に限りませんがSET命令はINTを含んでいて座標指定ができるのです。
WOPEN	10 WOPEN 20 PRINT/T X,X$	カセットテープに数値変数やストリング変数の内容を書き出すために、データ専用のファイルとしてオープンします。

	30 WOPEN "COST" 40 PRINT/T X,X$	上記と同様に変数のもつデータを"COST"というファイル名をつけて記憶させるための準備です。
ROPEN	10 ROPEN 20 INPUT/T Y,Y$	カセットテープに書き出された数値変数やストリング変数のデータを読み込むために、データ専用のファイルをオープンします。
	30 ROPEN "COST" 40 INPUT/T Y,Y$	上記と同様に"COST"というファイル名のついたデータ専用のファイルを捜して、変数のデータを読み込むための準備です。
CLOSE	50 CLOSE	カセットテープをデータ専用のファイルとして、使用することの終了を宣言します。WOPEN、ROPEN命令を使ったら、必ずファイルはクロースされなければなりません。
INPUT	10 INPUT A 20 INPUT A$	キーボードから変数Aに対する数値を入力します。 キーボードからストリング変数A$に対するストリングを入力します。
	←ストリング 30 INPUT "VALUE?";A	キーボードから入力する前に、ストリングのVALUE?を表示させます。ストリングと変数の区切りはセミコロン「;」を使います。
	40 INPUT A,A$,B,B$	数値変数やストリング変数はコンマ「,」で区切れば混合して使用できますが、入力する際には変数の型に合わせる必要があります。
INPUT/T	10 ROPEN 20 INPUT/T A 30 INPUT/T A$ 40 INPUT/T A,A$,B,B$	INPUTと同様の機能を、カセットテープから読み込みます。但しROPEN命令でオープンされたデータ専用のファイルがなければエラーになります。
GET	10 GET A	キーボードから変数Aに対して、1文字の数値を入力します。そのときキーが押されていないと、0が入力されます。
	20 GET A$	キーボードからストリング変数A$に対して、1個のストリングを入力します。そのときキーが押されていないと、A$は空になります。
PRINT	10 PRINT A 20 PRINT A$ 30 PRINT A;A$,B;B$	テレビ画面に変数Aの値を表示します。テレビ画面にストリング変数A$の文字列を表示します。 数値変数とストリング変数を混合して使用できます。また区切りでセミコロンが使われると、スペースなしで続けて表示され、コンマが使われると次の表示位置(10文字ごとの区切り)から表示されます。
	40 PRINT "COST=";A	クォーテーションマーク「"」で囲まれたストリングはその内容がそのまま表示されます。
	50 PRINT	PRINT命令だけの場合は、行替えになります。
PRINT/T	10 PRINT/T A 20 PRINT/T A;A$,B;B$ 30 PRINT/T "COST=";A	PRINTと同様の機能を、カセットテープに対して実行します。但し、WOPEN命令でオープンされたデータ専用のファイルがなければエラーになります。
PRINT/P	10 PRINT/P A 20 PRINT/P A;A$,B;B$ 30 PRINT/P "COST=";A	PRINTと同様の機能を、オプションのプリンターに対して実行します。プリンターがついていないときはエラーになります。
SIZE	PRINT SIZE	現在コンピュータ内にあるメモリーで、未使用のバイト数を表示します。
TI$	TI$="102030" 10 TI$="102030"	内蔵の時計を午前10時20分30秒にセットします。クォーテーションマーク「"」内には6文字の数字を用います。

	20 PRINT TI$	内蔵された時計の現在時刻を表示します。
CURSOR	80 CURSOR 25,15 90 PRINT "ABC"	テレビ画面の任意の箇所にカーソルを移動させるのがCURSORコマンドです。X軸方向には、左端から0～39、Y軸方向には上端から0～24の数値または変数で位置を指定します。左の例では、ストリング"ABC"は、画面の左端から26番目、上端から16番目のドット位置から表示されます。
TAB	10 PRINT TAB(X);A	テレビ画面の左端から数えてX＋1字目に変数Aの値を表示します。
	20 PRINT TAB(5);A$	テレビ画面の左端から数えて6字目からストリング変数A$の文字列を表示します。
SPC	10 PRINT SPC(X);A	カーソルの位置からX個のスペースを表示し、そののち変数Aの値を表示します。
	20 PRINT SPC(5);A$	カーソルの位置から5個のスペースを表示し、そののちストリング変数A$の文字列を表示します。
DIM		添字つき変数を使う場合には、このDIM(dimensionの略)文で、添字の最大を宣言しておかなければなりません。添字は0から、最大255まで使えます。
	10 DIM A(20)	添字つき変数A( )について、配列をA(0)からA(20)まで21個用意します。
	20 DIM B(79,79)	2重添字つき変数B( )について、配列をB(0、0)からB(79,79)まで6400個用意します。
	40 DIM C1$(10)	添字つきストリング変数C1$( )について、配列をC1$(0)からC1$(10)まで11個用意します。
	50 DIM K$(7,5)	2重添字つきストリング変数K$( )について、配列を、K$(0,0)からK$(7,5)まで48個用意します。
STOP	200 STOP	プログラムの実行をやめて、命令待ちとなります。ここでCONT命令を与えると、プログラムを続行します。
END	999 END	プログラムの最後を表わします。プログラムの実行をやめますが、CONT命令を与えると、さらに先のプログラムを実行します。
REM	100 REM オセロゲ゛ーム	REMは注釈文であり、プログラム実行の際無視されます。
INP		I／Oポート番号を指定して、そのポート上にあるデータの読み出しを行います。
	10 INP #12,A 20 PRINT A	文番号10で、I／Oポート番号12（10進）にあるデータを、変数Aに読み出します。
OUT		外部デバイスにデータを送るため、I／Oポート番号を指定して、データを出力します。
	10 B=A*2 20 OUT #13,B	文番号20で、変数Bの値をI／Oポート番号13の出力ポートへ出力します。
		（外部デバイスとの接続は、インタフェースユニットを介して行われます。INP、OUTコマンドの詳細は、インタフェースユニットおよびユニバーサルカードのマニュアルによって下さい。）

機械語とのリンク命令

命令	例	説明
LIMIT	10 LIMIT 49151 (←10進数) LIMIT 49151	BASICプログラムで使用するエリアを、49151番地（16進で$BFFF）に制限する命令です。
	30 LIMIT $BFFF (←16進数) LIMIT $BFFF	BASICプログラムで使用するエリアを、16進番地$BFFFに制限するものです。16進表現を行う場合は、このように"$"マークを用います。
	50 LIMIT $9FFF 60 LOAD "S-R 1"	機械語プログラム"S-R 1"が、ローディングアドレス$A000以上のものであれば、左のプログラムによって、機械語リンクエリア内に、カセットテープファイルから読み込まれます。
	90 LIMIT MAX LIMIT MAX	BASICプログラムで使用するエリアを、メモリの最大に設定します。
PEEK	20 A=PEEK(40960) (←10進数)	10進番地40960にストアされているデータを10進数に直して変数Aに代入します。
	20 A=PEEK($A000) (←16進数)	16進数を使って上記と同じ操作を実行させます。
	40 B=PEEK(AD)	変数ADで指定される10進番地にストアされているデータを10進数に直して変数Bに代入します。 （BASICエリアの番地を指定するとつねに32が読み出されます。）
POKE	70 POKE 45056,175 (←10進数)	10進番地45056にデータ175（10進表現）をセットします。
	70 POKE $B000,175 (←16進数 ←10進数)	16進数を使って上記と同じ操作を実行させます。
	100 POKE AD,DA	変数ADで指定する番地に、変数DAで示される値（0~255の範囲）をセットします。
USR	300 USR(49152) (←10進数)	10進番地49152にプログラムのコントロールを移します。これは機械語のCALL命令と同じ機能を持っています。従って、機械語プログラムのRET命令（10進コードで201）によってBASICプログラムのUSRコマンドを行なった次のステートメントへ戻ります。
	300 USR($C000) (←16進数)	16進数を使って上記と同じ操作を実行させます。
	400 USR(AD)	変数ADで指定する10進番地をCALLします。
	50 USR($A000,K$)	USR関数で、アドレスにつづけて、コンマを置き、ストリング変数を1個指定すると、DEレジスタの値は、該当ストリング（この場合K$）の入っている先頭番地となり、BCレジスタの値はストリング長となって、USR（$A000）を実行します。（この形の命令は、カラーディスプレイコントロールでも使われます。）

ストリング処理命令

LEFT$	300 A$=LEFT$(X$, n)	ストリング変数X$の最初からn文字目までを、ストリング変数A$に代入します。nは定数でも、変数、数式でもかまいません。
RIGHT$	600 B$=RIGHT$(X$, n)	ストリング変数X$の右からn文字を、ストリング変数B$に代入します。
MID$	900 C$=MID$(X$, m, n)	ストリング変数X$の第m文字目からn文字を、ストリング変数C$に代入します。
LEN	100 LX=LEN(X$) 110 LS=LEN(X$+Y$)	ストリング変数X$の文字の長さ（文字数）を、変数LXに代入します。 ストリング変数X$とY$の文字の長さの和を、変数LSに代入します。
ASC	200 A=ASC(X$)	ストリング変数X$の最初の文字のASCiiコード（10進数）の値を、変数Aに代入します。
CHR$	220 X1$=CHR$(A)	ASC関数の逆で、変数Aの値に等しいASCiiコードの文字（キャラクタ）をストリング変数X1$に代入します。Aは定数、変数、数式いずれでもかまいません。
VAL	400 I=VAL(N$)	ストリング変数N$の数字ストリングを、そのまま数値として変数Iに代入します。
STR$	440 N$=STR$(I)	VAL関数の逆で、変数Iの数値をそのままストリングとして、ストリング変数N$に代入します。

関　数

ABS	100 A=ABS(X)	変数Xの値の絶対値｜X｜を変数Aに代入します。カッコ内は定数、数式でもかまいません。 （例）　ABS(−3)=3 ABS(12)=12
SGN	100 A=SGN(X)	変数Xの値についてX<0のとき−1を、X=0のとき0を、X>0のとき、1を変数Aに代入します。カッコ内は定数、数式でもかまいません。 （例）　SGN(0.4)=1 SGN(0)=0 SGN(−400)=−1
INT	100 A=INT(X)	変数Xの値について、Xを越えない最大の整数を求めて変数Aに代入します。カッコ内は定数、数式でもかまいません。 （例）　INT(3.87)=3 INT(0.6)=0 INT(−3.87)=−4
SIN	100 A=SIN(X) 110 A=SIN(30*π/180)	変数Xの値（ラジアン）について、sinXの値を求め変数Aに代入します。カッコ内は定数、数式でもかまいません。ラジアンと度の関係は $1度=\frac{\pi}{180}$ラジアン ですから、たとえばsin30°の値を変数Aに代入するには文番号110のようにします。
COS	200 A=COS(X) 210 A=COS(200*π/180)	変数Xの値（ラジアン）について、cosXの値を求め変数Aに代入します。カッコ内は定数、数式でもかまいません。度で計算するにはSIN関係と同様の方法を使います。文番号210はcos200°の値を変数Aに代入する命令文です。

TAN	300 A=TAN (X) 310 A=TAN (Y＊π／180)	変数Xの値（ラジアン）について、tanXの値を求め変数Aに代入します。カッコ内は定数、数式でもかまいません。度で計算するにはＳＩＮ関数と同様の方法を使います。文番号３１０はtanY°の値を変数Aに代入する命令文です。
ATN	400 A=ATN (X) 410 A=180／π＊ATN (X)	変数Xの値について、$\tan^{-1}$Xの値（ラジアン）を求め変数Aに代入します。カッコ内は定数、数式でもかまいません。計算結果は$-\frac{\pi}{2}$と$\frac{\pi}{2}$の間の値となります。文番号４１０は$\tan^{-1}$Xの値を度にして変数Aに代入する命令文です。
SQR	100 A=SQR (X)	変数Xの値について、$\sqrt{X}$の値を求めて変数Aに代入します。カッコ内は定数、数式でもかまいませんが、正または0の値でなければなりません。
EXP	100 A=EXP (X)	変数Xの値について、e^Xの値を求めて変数Aに代入します。カッコ内は定数、数式でもかまいません。
LOG	100 A=LOG (X)	変数Xの値について、常用対数$\log_{10}$Xの値を求めて変数Aに代入します。カッコ内は定数、数式でもかまいませんが、正の値でなければなりません。
LN	100 A=LN (X) 110 A=LOG (X) ／LOG (Y) 120 A=LN (X) ／LN (Y)	変数Xの値について、自然対数$\log_e$Xの値を求めて変数Aに代入します。カッコ内は定数、数式でもかまいませんが、正の値でなければなりません。 対数の底がYのときの対数$\log_Y$Xを求めるには文番号１１０または文番号１２０によって求められます。
RND	100 A=RND (0) 110 B=RND (−3) 200 A=RND (1) 210 B=RND (10)	0.00000001から0.99999999までの値をとる乱数を発生する関数です。カッコ内に0または負の整数を与える場合と、正の整数を与える場合とで、２通りの処理が行われます。 文番号１００または１１０のようにカッコ内に0または負の整数を与えると乱数発生のイニシャライズが行なわれて、いつもある特定の数値を発生してAにもBにも同じ値が代入されます。 文番号２００または２１０のようにカッコ内に正の整数を与えるとRND関数を使うたびに、順次0.00000001から0.99999999までの間の値をとる乱数値を発生しますが、この場合カッコ内に与える正の整数の値には無関係です。

算術演算子

左端の白ヌキの数字は計算の優先順位、更に優先されるのはカッコ（　）内の計算です。

＝	10 A=X+3 （代入） 20 B=π	Xの変数値に3を加えた数値をAに代入します。 変数Bにπ(3.1415927)の数値を代入します。
❶ ↑	10 A=X↑Y （べき乗）	変数AにX^Yの計算結果を代入します。 （但しX↑YでXが負数のとき、Yが整数でなければエラーとなります）
❷ −	10 A=−B （負号）	0−Bは減算ですが、−Bの「−」は負号であることに注意して下さい。
❸ ＊	10 A=X＊Y （乗算）	変数AにXとYの数値の乗算結果を代入します。
❸ ／	10 A=X／Y （除算）	変数AにXとYの数値の除算結果を代入します。
❹ ＋	10 A=X+Y （加算）	変数AにXとYの数値の加算結果を代入します。
❹ −	10 A=X−Y （減算）	変数AにXとYの数値の減算結果を代入します。

論理演算子

記号	例	説明
=	`10 IF A=X THEN…`	変数AとXの数値が等しいならば、THEN以降の命令を実行します。
	`20 IF A$="XYZ" THEN…` （等しい）	ストリング変数A$の内容がストリングXYZであれば、THEN以降の命令を実行します。
`<>` または`><`	`10 IF A<>X THEN…` （等しくない）	変数AとXの数値が等しくないならば、THEN以降の命令を実行します。
`>=` または`=>`	`10 IF A>=X THEN…` （より大きいか等しい）	変数AがXより大きいか等しいならば、THEN以降の命令を実行します。
`<=` または`=<`	`10 IF A<=X THEN…` （より小さいか等しい）	変数AがXより小さいか等しいならば、THEN以降の命令を実行します。
*	`10 IF(A>X)*(B>Y) THEN…` （論理積）	変数AがXより大きく、かつ変数BがYより大きいならば、THEN以降の命令を実行します。
+	`10 IF(A>X)+(B>Y) THEN…` （論理和）	変数AがXより大きいか、または変数BがYより大きいならば、THEN以降の命令を実行します。

その他のシンボル

記号	例	説明
?	`200 ?"A+B=";A+B` `210 PRINT"A+B=";A+B`	PRINTの代わりに用いることができます。したがって文番号200と210は同等です。
:	`220 A=X:B=X↑2:PRINT A,B`	命令文の区切りを表わす記号で、多重命令に使用します。文番号220の多重命令には、3つの命令文が置かれています。
;	`230 PRINT"AB";"CD";"E"`	PRINTを続けて実行します。文番号230ではテレビ画面上に、「ABCDE」とスペースを空けずに続けて表示されます。
	`240 INPUT"X=";X$`	テレビ画面に「X=」と表示し、ストリング変数X$のキー入力を待ちます。
,	`250 PRINT"AB","CD","E"`	タブレーションをつけてPRINTを実行します。文番号250ではテレビ画面上にまずABと表示し、次にAから10文字右の場所よりCDと表示し、次にCから10文字右の場所にEと表示されます。
	`260 DIM A(20),B(30)`	変数の区切りに用いた例です。
"	`270 A$="SHARP BASIC"` `280 PRINT"*";A$;"*"`	" "内がストリングであることを示します。
$	`290 A1$=LEFT$(A$,5)` `300 A2$="MZ-80K"`	ストリング変数であることを示します。
	`350 LIMIT $A000` `370 POKE $A010,201`	16進数であることを示します。
π	`400 I=SIN(X*π/180)`	円周率3.1415927をπで表わします。

エラー表示

コンピュータの動作中にエラーが発生した場合、次の形式でコンピュータよりテレビ画面上に、エラーメッセージが送られます。

① エラーの種類　ERROR	（直接命令を実行中でのエラー）
② エラーの種類　ERROR　IN　文番号	（プログラム実行中でのエラー）

①はとくにプログラムの作成中か修正中に発生し、文番号に無関係なエラーとみなされる場合
②はとくにプログラムの実行中において、文番号××××に発生したエラーとみなされる場合

さらに、エラーの種類は次のものがあります。

SYNTAX
プログラムの文法上に誤まりがある場合

MEMORY
プログラムとして使える記憶容量を越えた場合

DATA
コンピュータが扱えるデータの範囲を越えた場合

MISMATCH
数値を扱うための変数とストリングを扱うための変数を混合して使った場合

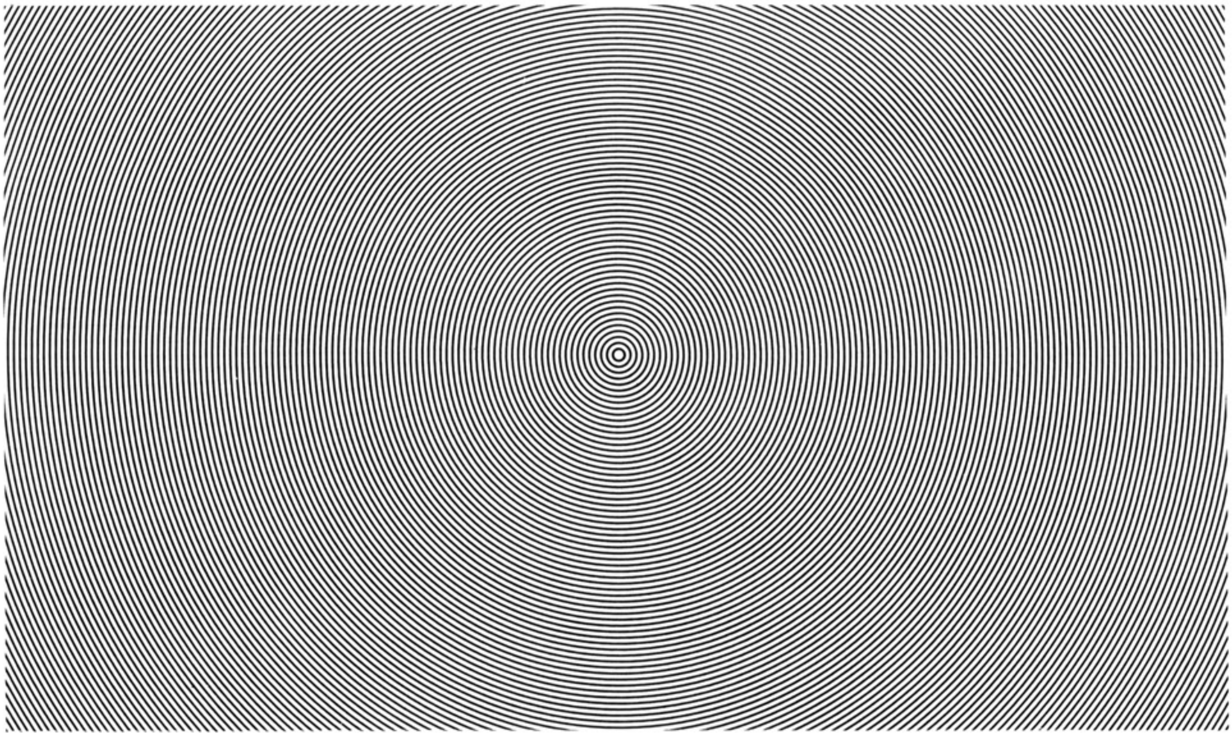
16FOR
FOR−NEXTのループで、16重以上のループが重なった場合

16GOSUB
GOSUBが多すぎて、16重以上にGOSUBを重ねて使った場合

6FN
DEF　FNの関数定義では、その中で更に別の関数定義を使えるが、この多重定義の範囲が6重を越えた場合

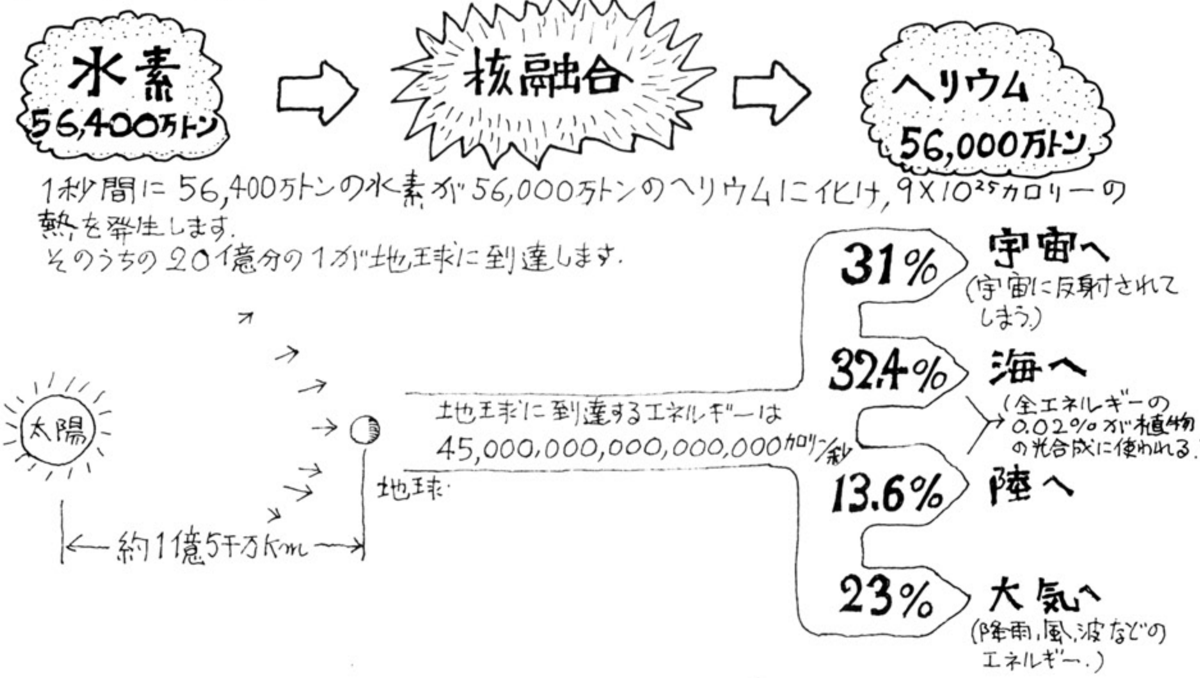
CONT
STOP命令、END命令を実行したとき、又は BREAK をかけてプログラムの実行を中止させたとき、CONT命令で中止したところから、実行を続行させることができます。これ以外でCONT命令を使ったり、STOPしたあとで何らかのプログラムを編集しなおしたりすると、このエラーになります。
（PRINT A やLIST命令のような直接命令は使ってもさしつかえありません）

毎秒400万トンずつやせ細る太陽



物質が核反応によってエネルギーに変わるとき、その分だけ質量が減少します。太陽は水素を核融合反応によってヘリウムに変え、1秒当たり 9×10^{25} カロリーもの熱を発生します。そして、そのときに消失する質量は 400 万トン。太陽は、自分の身を細らせながら、私たち生物を育てているのです。とはいうものの、生物の住む唯一の惑星と考えられる地球を照射するエネルギーは、その約20億分の1、しかもその 1/3 は宇宙に反射してしまいます。今、地球上では人びとが有効に使えるエネルギーが不足しているために皆が悩んでいますが、何とももったいない話ですね。

太陽の核融合反応のしくみ



人は
昔から
道具を使った。
そうだ！　コンピュータも
自動車のような道具なんだ
でも何て頭のいい道具なんだろ。
この本をあなたの友だちに見せて下さい
これからの勉強法だって？………
自分でどんどんプログラムを
作ってみることが
最良の
勉強
❢

ディスプレイコード

MZ−80シリーズのディスプレイコードを下表に示します。コードは10進数によっています。たとえば、POKE命令でビデオRAMエリア（10進番地の53248～54247）内にこのコードを転送すると、テレビ画面の該当位置にその文字が表示されます。たとえば、POKE　53248，1　を実行させると、テレビ画面の左上隅に"A"が表示されます。なお表中で四角カッコで囲んでいないカナ文字は小文字を表わします。

コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字
0	SP	37	5	74		111		148	カ	185	・	222	£
1	A	38	6	75		112		149	ナ	186	ェ	223	
2	B	39	7	76		113		150	ヒ	187	ッ	224	
3	C	40	8	77		114		151	テ	188	゛	225	
4	D	41	9	78		115		152	サ	189	゜	226	
5	E	42	−	79	:	116		153	ン	190	ォ	227	
6	F	43	=	80	↑	117		154	ツ	191	ー	228	
7	G	44	;	81	<	118		155	ロ	192		229	
8	H	45	/	82	[	119		156	ケ	193	↓	230	
9	I	46	.	83	♥	120		157	「	194	↑	231	
10	J	47	,	84	]	121		158	ァ	195	→	232	
11	K	48		85	@	122		159	ャ	196	←	233	
12	L	49		86		123		160	ワ	197		234	
13	M	50		87	>	124		161	ヌ	198		235	
14	N	51		88		125		162	フ	199		236	
15	O	52		89	\	126		163	ア	200		237	
16	P	53		90		127		164	ウ	201		238	
17	Q	54		91		128	SP	165	エ	202		239	
18	R	55		92		129	チ	166	オ	203		240	SP
19	S	56		93		130	コ	167	ヤ	204		241	
20	T	57		94		131	ソ	168	ユ	205	¥	242	
21	U	58		95		132	シ	169	ヨ	206	●	243	
22	V	59		96	π	133	イ	170	ホ	207	☺	244	
23	W	60		97	!	134	ハ	171	ヘ	208	日	245	
24	X	61		98	"	135	キ	172	レ	209	月	246	
25	Y	62		99	#	136	ク	173	メ	210	火	247	
26	Z	63		100	$	137	ニ	174	ル	211	水	248	
27		64	SP	101	%	138	マ	175	ネ	212	木	249	
28		65	♠	102	&	139	ノ	176	ム	213	金	250	
29		66		103	'	140	リ	177	」	214	土	251	
30		67	■	104	(	141	モ	178	ィ	215	生	252	
31		68	♦	105	)	142	ミ	179	ュ	216	年	253	
32	0	69	←	106	+	143	ラ	180	ヲ	217	時	254	
33	1	70	♣	107	*	144	セ	181	、	218	分	255	
34	2	71	●	108		145	タ	182	ゥ	219	秒		
35	3	72	○	109		146	ス	183	ョ	220	円		
36	4	73	?	110		147	ト	184	。	221	¥		

（注）表中で SP は空白（スペース、ブランク）を表わすものとします。

特殊文字コードとメモリーマップ

特殊文字コード

次に示す特殊文字コードは　"　"（引用符）をつけてPRINT命令を実行したとき、カーソルの移動、ホーム設定、画面クリアの機能をもちます。MZ－80シリーズにはこのコードが6つあります。コードは10進数によります。

コード	記号文字	機　　能	キ　ー
193	↓	下に1文字分カーソル移動	↓
194	↑	上に1文字分カーソル移動	↑
195	→	右に1文字分カーソル移動	→
196	←	左に1文字分カーソル移動	←
197	H	画面左上すみにカソール移動(ホーム)	HOME
198	C	画面クリアとともにホーム	CLR

メモリーマップ

RAM 48Kバイトフル実装として、BASICを搭載した場合、概略下記のエリア分割となります。



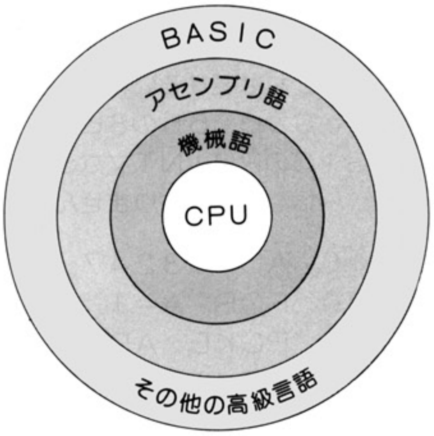
アスキーコード表

MZ－80シリーズのアスキーコードを下表に示します。BASICの　CHR$(n)、ASC(X$)による文字とコードの変換はこれに相当します。コードは10進数によっています。なお表中で四角カッコで囲んでいないカナ文字は小文字を表わします。また CR はキャリッジリターンコード、17～22まではカーソルコントロールのためのコードです。

コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字
0		37	%	74	J	111		148	エ	185	ル	222	
1		38	&	75	K	112	日	149	オ	186	レ	223	
2		39	'	76	L	113	月	150	カ	187	ロ	224	
3		40	(	77	M	114	火	151	キ	188	ワ	225	♠
4		41	)	78	N	115	水	152	ク	189	ン	226	
5		42	*	79	O	116	木	153	ケ	190	゛	227	
6		43	+	80	P	117	金	154	コ	191	゜	228	
7		44	,	81	Q	118	土	155	サ	192	SP	229	
8		45	−	82	R	119	生	156	シ	193		230	
9		46	.	83	S	120	年	157	ス	194		231	
10		47	/	84	T	121	時	158	セ	195		232	
11		48	0	85	U	122	分	159	ソ	196		233	
12		49	1	86	V	123	秒	160	タ	197		234	
13	CR	50	2	87	W	124	円	161	チ	198		235	
14		51	3	88	X	125	¥	162	ツ	199		236	
15		52	4	89	Y	126	£	163	テ	200	■	237	
16		53	5	90	Z	127		164	ト	201		238	
17	↓	54	6	91	[	128	SP	165	ナ	202		239	
18	↑	55	7	92	\	129	。	166	ニ	203		240	
19	→	56	8	93	]	130	「	167	ヌ	204		241	●
20	←	57	9	94	↑	131	」	168	ネ	205		242	
21	H	58	:	95	←	132	、	169	ノ	206		243	♥
22	C	59	;	96		133	・	170	ハ	207		244	
23		60	<	97		134	ヲ	171	ヒ	208		245	
24		61	=	98		135	ァ	172	フ	209		246	
25		62	>	99		136	ィ	173	ヘ	210		247	○
26		63	?	100		137	ゥ	174	ホ	211		248	♣
27		64	@	101		138	ェ	175	マ	212		249	
28		65	A	102		139	ォ	176	ミ	213		250	♦
29		66	B	103	●	140	ャ	177	ム	214		251	
30		67	C	104	☺	141	ュ	178	メ	215		252	
31		68	D	105		142	ョ	179	モ	216		253	
32	SP	69	E	106		143	ッ	180	ヤ	217		254	
33	!	70	F	107		144	ー	181	ユ	218		255	π
34	"	71	G	108		145	ア	182	ヨ	219			
35	#	72	H	109		146	イ	183	ラ	220			
36	$	73	I	110		147	ウ	184	リ	221			

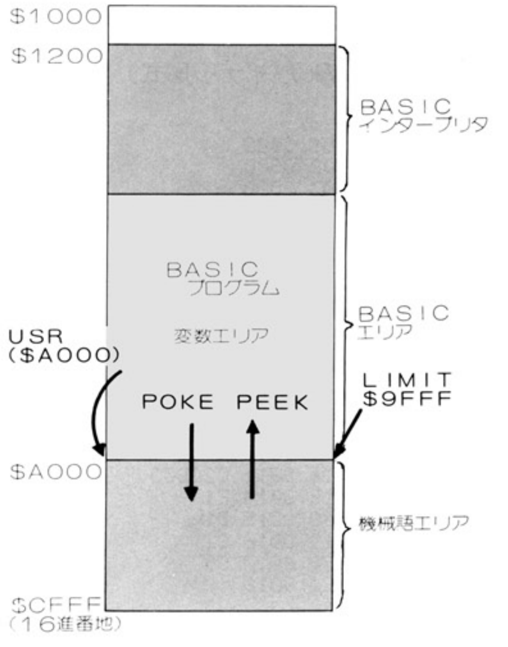
（注）表中で SP は空白（スペース、ブランク）を表わすものとします。

機械語とのリンク



MZ−80シリーズBASICには、機械語プログラムのコントロール命令があり、BASICプログラムとリンクさせることができます。機械語プログラムとは、BASICのような高級言語、即ち、人間にとってわかりやすい言語によるプログラム言語とはちがって、コンピュータの心臓部である中央処理装置（CPU）に直接命令を与えるものであり、言語というより機械コードといった性質のものです。
コンピュータにとって、プログラミング言語とは、いくつかのレベルによって階層的なものとして捉えることができ、BASICなどの高級言語、アセンブリ言語、機械語は、およそ右図に示したような階層をなして、中央処理装置との位置づけがなされています。BASICプログラムなどでは、アセンブラが行うべき部分や機械語に変換される部分などが、BASIC内部で自動的に行なわれ、そうした操作を私たちがじかに行わなくても済むようになっているのです。
しかし逆に、私たちが、CPUをじかにコントロールして、たとえば高速の処理を行わせたい場合や、BASIC命令では実現できないような特殊なコントロールを実行したい場合が考えられます。そんな時には、オプションのソフトウェア、「SYSTEM PROGRAM」や「MACHINE LANGUAGE」等を使うことによって、自在に、アセンブラプログラミングや機械語プログラミングができます。そうした多様な応用性が、MZ−80シリーズクリーンコンピュータの大きな特質ですが、ここで解説する、BASICプログラムと機械語とのリンクは、両者が共存してそれぞれの特質を生かしたプログラミングが可能なことを示すものです。

BASICプログラムと機械語プログラムのリンクは、一般的には、「SYSTEM PROGRAM」または「MACHINE LANGUAGE」で作成された機械語プログラムファイルを、BASICのLIMIT命令を使って確保した機械語プログラムエリアにローディングして行われます。そして、BASIC命令、USR(　)を使って、機械語プログラムをサブルーチンとしてCALLするのです。



それに対して、BASIC単独で使用する場合には、POKE命令、PEEK命令を用いて、直接のメモリアクセスを行うことになります。すなわち、POKE命令による、機械語プログラムの機械語エリアへの書き込み、PEEK命令による、機械語エリアからのデータ読み出しを行いながらプログラムを作成してゆき、出来上ったら、USR(　)命令を使ってリンクするわけです。
右図は、RAM48Kバイトフル装備の場合に、LIMIT $9FFFを行って、BASICエリアと機械語エリアとを分離しておいて、POKE命令によるデータ書き込み、PEEK命令によるデータ読み出し、そして、USR(　)によるプログラムのリンキングの大まかなようすを示したものです。

POKE命令、PEEK命令で扱う機械語プログラムデータとは、後に示す、「Z80命令表」にある命令のコード（2進コード）を10進数に直したものです。さあ、これらのコントロールを、実際に行ってみることにしましょう。

機械語プログラムの作成

──例題──

テレビ画面いっぱい、MZ－80シリーズにあるキャラクタの全部を使って埋めつくすことを考えてみます。これは要するに、キャラクタジェネレータ内の全ての文字を表示させる訳であり、ビデオRAMに、0から255の8ビットデータの全てを順次セットすればよいことになります。すなわち、BASICのPRINT文などは使えないことになり、何らかの方法で、ビデオRAMに直接データをセットしなくてはなりません。これはたとえば、

```
10  AD=53247
20  FOR  A=1  TO  4  :FOR  C=0  TO  255  :AD=AD+1
30  POKE  AD,C
40  NEXT  C  :NEXT  A
```

といった、POKE文を使って実行する方法が考えられます。このプログラムは、つまりこういうことです。ビデオRAMは、先のメモリーマップの図からわかるように、$D000＝53248番地からとなっており、テレビ画面に、キャラクタジェネレータを通して接続されています。そこに、0～255のデータをインクリメントしながら順番にセットして行っているわけで、つまり

POKE AD, C … 変数ADに相当するアドレスに、変数Cに相当するデータをセットする。

という働きが、POKE命令にあることがわかると思います。さて、考え方を変えて、今度は機械語プログラムを機械語エリアに作成して、そのサブルーチンをBASICから呼び出して以上の動作を行わせるようにしてみます。その一例は、次のようになります。ここではBASICエリアは、$7FFF＝32767に限定しています。

```
10 PRINT "C":LIMIT $7FFF:GOSUB 100:USR($8000):END
100 FOR AD=32768 TO 32791 :READ D
110 POKE AD,D :NEXT AD :RETURN
120 DATA 197,213,229,22,0,33,0,208,1,232,3,114,35
130 DATA 20,11,120,177,194,11,128,225,209,193,201
```

DATA文中にあるものが機械語プログラムで、それを、$8000～$8017番地にPOKE命令によって書き込んだ後に、USR（$8000）を行っているのです。このプログラムは、SYSTEM PROGRAMのマクロアセンブラのアセンブルリストで示すと下のようになります。（リロケータブルバイナリ形式）

```
01 0000                 ;
02 0000                 ;  DISPLAY ALL CHARACTERS
03 0000                 ;
04 0000 C5              START:  PUSH  BC          ;PUSH REGISTERS
05 0001 D5                      PUSH  DE
06 0002 E5                      PUSH  HL
07 0003 1600                    LD    D,0H        ;D<--DISPLAY CODE
08 0005 2100D0                  LD    HL,D000H    ;HL<--TOP OF V-RAM
09 0008 01E803                  LD    BC,1000     ;BC<--1000(DECIMAL)
10 000B 72              STORE:  LD    (HL),D      ;SET CODE TO V-RAM
11 000C 23                      INC   HL
12 000D 14                      INC   D
13 000E 0B                      DEC   BC
14 000F 78                      LD    A,B
15 0010 B1                      OR    C
16 0011 C20B00                  JP    NZ,STORE
17 0014 E1                      POP   HL          ;POP REGISTERS
18 0015 D1                      POP   DE
19 0016 C1                      POP   BC
20 0017 C9                      RET               ;RETURN TO BASIC TEXT
21 0018                         END
```

（このリストは、ドットプリンタMZ－80P3による出力例です。）

USR命令と機械語プログラムファイル

USR(　)命令は、(　)内に示したアドレスの機械語サブルーチンプログラムをCALL（コール）する命令です。従って、今の例でも機械語プログラムの終わりには　201＝RETの10進数コード、が置かれていて、それによって、もとのBASICテキストへRETURNしています。

これに対して、USR(address、string）という形のUSR命令もあり、これはUSR(address)の働きとともに、次の条件で、ストリング変数の内容をレジスタ値にもってCALLが行われます。

DEレジスタの値………該当ストリング変数の内容がセットされている場所の先頭番地
BCレジスタの値………該当ストリング変数の文字数（最大255）

たとえば、A$＝"MZ−80SERIES　CLEAN　COMPUTERS"であるとき、USR（$8000、A$）を実行したとすると、コントロールは、$8000番地へ移され、その時、DEレジスタの値は、A$の先頭、すなわち、文字"M"のセットされている番地を指し示しており、BCレジスタの値は、A$の文字数、すなわち27＝LEN（A$）となっているのです。これによって、BASIC内で使用しているストリング変数を、機械語プログラムにリンクすることができることになります。

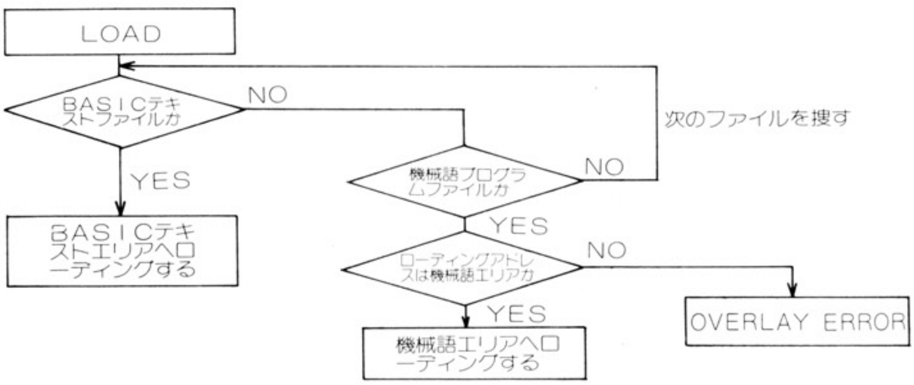
PEEK命令は、機械語エリア内の或る番地にセットされているデータを読み出すもので、たとえば

D＝PEEK（40000）

とすると、10進番地40000にあるデータを、変数Dに代入します。この場合、BASICエリア内の番地を指定すると常に32が読み出されます。

機械語エリア内のデータを、そのままファイルへSAVEすることはできません。それを行うには、たとえば、PEEK命令によってそのデータを変数に代入して来て、それをWOPEN命令、PRINT／T命令を用いてデータファイルに書き出す方法が考えられます。ファイルの読み込みは、ROPEN命令、INPUT／T命令を用いて、データを変数に読み込み、それをPOKE命令を使って機械語エリアへセットすることになります。

「SYSTEM　PROGRAM」または「MACHINE　LANGUAGE」によって作成した機械語プログラムファイルを使用する場合、LOAD命令によってプログラムのローディングを行うことができることは先に述べました。従って、LOAD命令は、BASICテキストファイルの読み込みにも、機械語プログラムファイルの読み込みにも使えるのです。LOAD命令は、次のような形式で実行されます。



テレビ画面の構成と特殊な制御命令

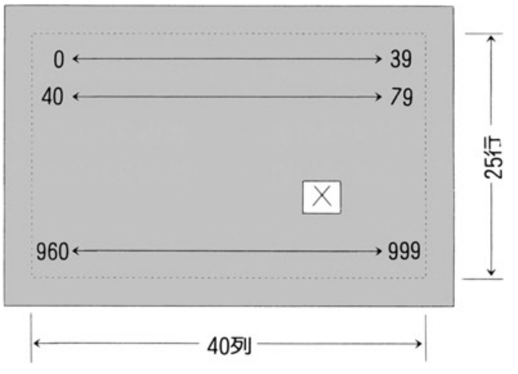
テレビ画面の構成

テレビ画面は右図のように２５行４０列の画素子から構成されています。１つの画素はさらに８×８ドットマトリックスによって構成されています。これらのドットの組合せで１つの画素子に、ある文字を表示しています。

テレビ画面上の画素子の位置は、ビデオRAMとして内部メモリーの５３２４８ ～５７３４３（１０進数番地）に割り当てられています。

テレビ画面の任意の位置に、ある文字を表示するにはカーソルなどを用いてキー入力すればよいのですが、次に示すようなPOKE 命令を用いることも可能です。

例えば任意の位置にある画素子のメモリー番地５３２４８からの変位Ｘ（０≦Ｘ≦９９９）を指定し、表示したい文字のディスプレイコードをＹ（０≦Ｙ≦２５５）とすれば、



POKE　53248+X，Y

の命令によりその文字がＸの位置に表示されます。

特殊な制御命令

MZ－８０シリーズBASICには、前記のPOKE命令を含め次のような特殊な制御命令があります。

POKE　53248+X，Y	；画素子Ｘの位置にディスプレイコードＹに相当する文字を表示する。
POKE　4509，0	；キーを押したときにエントリーベルが鳴る。
POKE　4509，1	；キーを押したときのエントリーベルをとめる。
USR（62）	；このステートメントを実行することにより、ベルが鳴る。




HELLO!
MY FRIEND
HELLO!
MY COMPUTER

I／O入出力命令

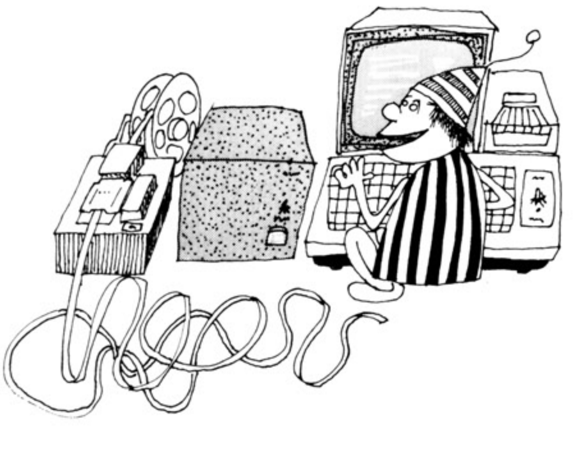
BASIC（SP−5030）のコマンド解説の最後に、INP、OUT命令の使い方を説明いたします。この2つの命令は、外部機器とのデータ入出力を行うためのもので、オプションのユニバーサルI／Oカードを介して、8ビットデータ（10進で、0〜255の範囲のコード）を入出力することができます。MZ−80シリーズは、プリンタ、フロッピーディスク等を含めて、外部機器とのインターフェイスは、オプションのインターフェイスユニット（MZ−80I／O）を用いて行います。各専用外部機器には、それぞれインターフェイスカードが付属になっており、それをインターフェイスユニットに挿入してコンピュータ本体と各機器とを接続するようになっています。ユニバーサルI／Oカード（MZ−80I／O1）は、文字通り、汎用I／Oであり、特定の機器のためのものでなく、ユーザの自由な利用のできるインターフェイスカードなのです。
ユニバーサルI／Oカード上には、ポートアドレス選択スイッチがついており、I／Oポート番号として、0〜239（16進の00H〜EFH）まで任意に決めることができます。240〜255（16進のF0H〜FFH）までは、各専用外部機器用ポートとして専用に使用されています。
INP、OUT命令は、このポートアドレスを指定して、次のような形で使用します。

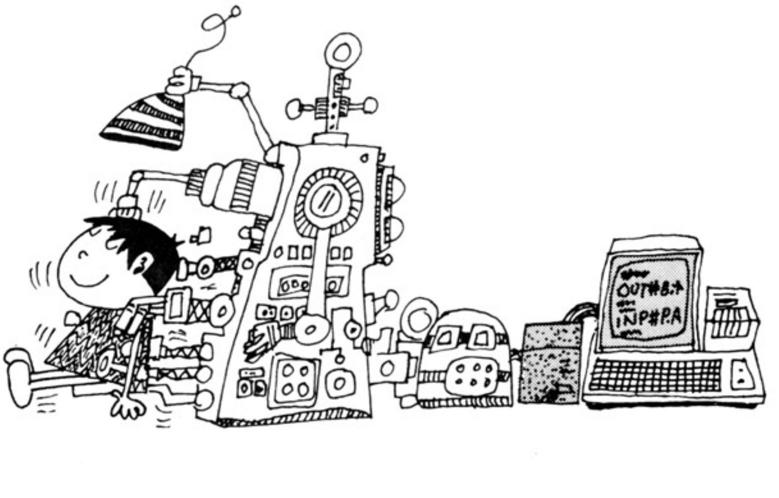
INP　#P，D　………ポートアドレスPの入力ポートより8ビットデータを入力し、それを10進変換した数値（0〜255）を、変数Dに代入する。

OUT　#P，D　………ポートアドレスPの出力ポートに変数Dの値（0〜255）を2進数に変換して出力する。

ユニバーサルI／Oカードの仕様（入出力レベル、使用ICなど）の詳細は、「UNIVERSAL INTERFACE CARD MANUAL」によってください。

ユニバーサルI／Oカードの使用によって、MZ−80シリーズクリーンコンピュータを、各種のユーザ機器に接続させることができますから、たとえば、X−Yプロッタや、PTR、PTP（紙テープリーダ、パンチャ）や、その他ミュージックシンセサイザなどをコントロールすることが可能です。
MZ−80クリーンコンピュータのベースシステム（プリンタ、フロッピーディスク、インターフェイスユニットの構成する基本的なシステム構成）に加えて、巾広い拡張性、発展性がMZ−80シリーズには随所に盛り込まれているのです。



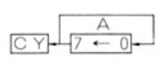
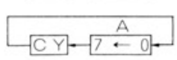
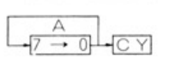
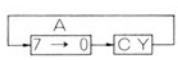
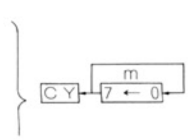


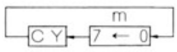
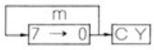
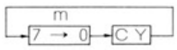
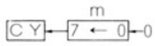
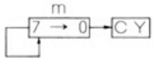
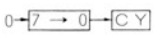
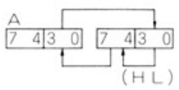
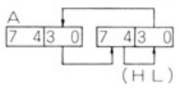
Z80 命令表

MZ－80シリーズは中央処理装置(CPU)としてZ80を用いていますので、参考のため、以下にその命令表を示しておきます。

ニーモニック	実行内容	命令コード 76 543 210
8ビットロードグループ		
LD r, r'	r←r'	01 r r'
LD r, n	r←n	00 r 110
		← n →
LD r, (HL)	r←(HL)	01 r 110
LD r, (IX+d)	r←(IX+d)	11 011 101
		01 r 110
		← d →
LD r, (IY+d)	r←(IY+d)	11 111 101
		01 r 110
		← d →
LD (HL), r	(HL)←r	01 110 r
LD (IX+d), r	(IX+d)←r	11 011 101
		01 110 r
		← d →
LD (IY+d), r	(IY+d)←r	11 111 101
		01 110 r
		← d →
LD (HL), n	(HL)←n	00 110 110
		← n →
LD (IX+d), n	(IX+d)←n	11 011 101
		00 110 110
		← d →
		← n →
LD (IY+d), n	(IY+d)←n	11 111 101
		00 110 110
		← d →
		← n →
LD A, (BC)	A←(BC)	00 001 010
LD A, (DE)	A←(DE)	00 011 010
LD A, (nn)	A←(nn)	00 111 010
		← n →
		← n →
LD (BC), A	(BC)←A	00 000 010
LD (DE), A	(DE)←A	00 010 010
LD (nn), A	(nn)←A	00 110 010
		← n →
		← n →
LD A, I	A←I	11 101 101
		01 010 111
LD A, R	A←R	11 101 101
		01 011 111
LD I, A	I←A	11 101 101
		01 000 111
LD R, A	R←A	11 101 101
		01 001 111
16ビットロードグループ		
LD dd, nn	dd←nn	00 dd0 001
		← n →
		← n →
LD IX, nn	IX←nn	11 011 101
		00 100 001
		← n →
		← n →
LD IY, nn	IY←nn	11 111 101
		00 100 001
		← n →
		← n →
LD HL, (nn)	H←(nn+1)	00 101 010
	L←(nn)	← n →
		← n →
LD dd, (nn)	dd_H←(nn+1)	11 101 101
	dd_L←(nn)	01 dd1 011
		← n →
		← n →
LD IX, (nn)	IX_H←(nn+1)	11 011 101
	IX_L←(nn)	00 101 010
		← n →
		← n →
LD IY, (nn)	IY_H←(nn+1)	11 111 101
	IY_L←(nn)	00 101 010
		← n →
		← n →
LD (nn), HL	(nn+1)←H	00 100 010
	(nn)←L	← n →
		← n →
LD (nn), dd	(nn+1)←dd_H	11 101 101
	(nn)←dd_L	01 dd0 011
		← n →
		← n →
LD (nn), IX	(nn+1)←IX_H	11 011 101
	(nn)←IX_L	00 100 010
		← n →
		← n →
LD (nn), IY	(nn+1)←IY_H	11 111 101
	(nn)←IY_L	00 100 010
		← n →
		← n →
LD SP, HL	SP←HL	11 111 001
LD SP, IX	SP←IX	11 011 101
		11 111 001
LD SP, IY	SP←IY	11 111 101
		11 111 001
PUSH qq	(SP−2)←qq_L	11 qq0 101
	(SP−1)←qq_H	
PUSH IX	(SP−2)←IX_L	11 011 101
	(SP−1)←IX_H	11 100 101
PUSH IY	(SP−2)←IY_L	11 111 101
	(SP−1)←IY_H	11 100 101
POP qq	qq_H←(SP+1)	11 qq0 001
	qq_L←(SP)	
POP IX	IX_H←(SP+1)	11 011 101
	IX_L←(SP)	11 100 001
POP IY	IY_H←(SP+1)	11 111 101
	IY_L←(SP)	11 100 001
エクスチェンジグループ、ブロック転送及び、サーチグループ		
EX DE, HL	DE↔HL	11 101 011
EX AF, AF'	AF↔AF'	00 001 000
EXX	(BC)↔(BC')	11 011 001
	(DE)↔(DE')	
	(HL)↔(HL')	
EX (SP), HL	H↔(SP+1)	11 100 011
	L↔(SP)	
EX (SP), IX	IX_H↔(SP+1)	11 011 101
	IX_L↔(SP)	11 100 011
EX (SP), IY	IY_H↔(SP+1)	11 111 101
	IY_L↔(SP)	11 100 011

ニーモニック	実行内容	命令コード 76 543 210
LDI	(DE)←(HL)	11 101 101
	DE←DE+1	10 100 000
	HL←HL+1	
	BC←BC−1	
LDIR	(DE)←(HL)	11 101 101
	DE←DE+1	10 110 000
	HL←HL+1	
	BC←BC−1	
	BC=0までくりかえす	
LDD	(DE)←(HL)	11 101 101
	DE←DE−1	10 101 000
	HL←HL−1	
	BC←BC−1	
LDDR	(DE)←(HL)	11 101 101
	DE←DE−1	10 111 000
	HL←HL−1	
	BC←BC−1	
	BC=0までくりかえす	
CPI	A−(HL)	11 101 101
	HL←HL+1	10 100 001
	BC←BC−1	
CPIR	A−(HL)	11 101 101
	HL←HL+1	10 110 001
	BC←BC−1	
	A=(HL)又はBC=0までくりかえす	
CPD	A−(HL)	11 101 101
	HL←HL−1	10 101 001
	BC←BC−1	
CPDR	A−(HL)	11 101 101
	HL←HL−1	10 111 001
	BC←BC−1	
	A=(HL)又はBC=0までくりかえす	
8ビット演算・論理グループ		
ADD A, r	A←A+r	10 [000] r
ADD A, n	A←A+n	11 [000] 110
		← n →
ADD A, (HL)	A←A+(HL)	10 [000] 110
ADD A, (IX+d)	A←A+(IX+d)	11 011 101
		10 [000] 110
		← d →
ADD A, (IY+d)	A←A+(IY+d)	11 111 101
		10 [000] 110
		← d →
ADC A, s	A←A+s+CY	[001]
SUB s	A←A−s	[010]
SBC A, s	A←A−s−CY	[011]
AND s	A←A∧s	[100]
OR s	A←A∨s	[110]
XOR s	A←A⊕s	[101]
CP s	A−s	[111]
INC r	r←r+1	00 r [100]
INC (HL)	(HL)←(HL)+1	00 110 [100]
INC (IX+d)	(IX+d)	11 011 101
	←(IX+d)+1	00 110 [100]
		← d →
INC (IY+d)	(IY+d)	11 111 101
	←(IY+d)+1	00 110 [100]

ニーモニック	実行内容	命令コード 76 543 210
		← d →
DEC m	m←m−1	[101]
一般目的の演算、及びCPUコントロールグループ		
DAA	加減算後のAの内容の10進補正を行う。	00 100 111
CPL	A←$\overline{A}$	00 101 111
NEG	A←$\overline{A}$+1	11 101 101
		01 000 100
CCF	CY←$\overline{CY}$	00 111 111
SCF	CY←1	00 110 111
NOP	何も実行されないがPCの内容はインクリメントされる	00 000 000
HALT	CPU停止	01 110 110
DI	IFF←0	11 110 011
EI	IFF←1	11 111 011
IM0	割り込みモード0にする。	11 101 101
		01 000 110
IM1	割り込みモード1にする。	11 101 101
		01 010 110
IM2	割り込みモード2にする。	11 101 101
		01 011 110
16ビット演算グループ		
ADD HL, ss	HL←HL+ss	00 ss1 001
ADC HL, ss	HL←HL+ss+CY	11 101 101
		01 ss1 010
SBC HL, ss	HL←HL−ss−CY	11 101 101
		01 ss0 010
ADD IX, pp	IX←IX+pp	11 011 101
		00 pp1 001
ADD IY, rr	IY←IY+rr	11 111 101
		00 rr1 001
INC ss	ss←ss+1	00 ss0 011
INC IX	IX←IX+1	11 011 101
		00 100 011
INC IY	IY←IY+1	11 111 101
		00 100 011
DEC ss	ss←ss−1	00 ss1 011
DEC IX	IX←IX−1	11 011 101
		00 101 011
DEC IY	IY←IY−1	11 111 101
		00 101 011
ローテイト・シフトグループ		
RLCA		00 000 111
RLA		00 010 111
RRCA		00 001 111
RRA		00 011 111
RLC r		11 001 011
		00 [000] r
RLC (HL)		11 001 011
		00 [000] 110

ニーモニック	実行内容	命令コード 76 543 210
RLC (IX+d)		11 011 101 11 001 011 ← d → 00 000 110
RLC (IY+d)		11 111 101 11 001 011 ← d → 00 000 110
RL m		010
RRC m		001
RR m		011
SLA m		100
SRA m		101
SRL m		111
RLD		11 101 101 01 101 111
RRD		11 101 101 01 100 111
ビットセット、リセット及びテストグループ		
BIT b, r	$Z \leftarrow \overline{r_b}$	11 001 011 01 b r
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	11 001 011 01 b 110
BIT b, (IX+d)	$Z \leftarrow \overline{(IX+d)_b}$	11 011 101 11 001 011 ← d → 01 b 110
BIT b, (IY+d)	$Z \leftarrow \overline{(IY+d)_b}$	11 111 101 11 001 011 ← d → 01 b 110
SET b, r	$r_b \leftarrow 1$	11 001 011 11 b r
SET b, (HL)	$(HL)_b \leftarrow 1$	11 001 011 11 b 110
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	11 011 101 11 001 011 ← d → 11 b 110
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	11 111 101 11 001 011 ← d → 11 b 110
RES b, m	$m_b \leftarrow 0$	10

ニーモニック	実行内容	命令コード 76 543 210
ジャンプグループ		
JP nn	PC←nn	11 000 011 ← n → ← n →
JP cc, nn	ccが真のとき、 PC ←nn	11 cc 010 ← n → ← n →
JR e	PC←PC+e	00 011 000 ← e-2 →
JR C, e	C=0のとき、 次の命令へ C=1のとき、 PC←PC+e	00 111 000 ← e-2 →
JR Z, e	Z=0のとき 次の命令へ Z=1のとき、 PC←PC+e	00 101 000 ← e-2 →
JR NC, e	C=1のとき、 次の命令へ C=0のとき、 PC←PC+e	00 110 000 ← e-2 →
JR NZ, e	Z=1のとき、 次の命令へ Z=0のとき、 PC←PC+e	00 100 000 ← e-2 →
JP (HL)	PC←HL	11 101 001
JP (IX)	PC←IX	11 011 101 11 101 001
JP (IY)	PC←IY	11 111 101 11 101 001
DJNZ e	B←B−1 B=0のとき、 次の命令へ B≠0のとき、 PC←PC+e	00 010 000 ← e-2 →
コール、リターングループ		
CALL nn	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ PC←nn	11 001 101 ← n → ← n →
CALL cc, nn	ccが真のとき、 CALL nnと同じ ccが偽のとき継続	11 cc 100 ← n → ← n →
RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP+1)$	11 001 001
RET cc	ccが真のとき RETと同じ ccが偽なら継続	11 cc 000
RETI	割り込みからの リターン	11 101 101 01 001 101
RETN	$\overline{NMI}$からのリターン	11 101 101 01 000 101
RST p	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	11 t 111

ニーモニック	実行内容	命令コード 76 543 210
入力・出力グループ		
IN A, (n)	A←(n)	11 011 011 ← n →
IN r, (C)	r←(C)	11 101 101 01 r 000
INI	(HL)←(C) B←B−1 HL←HL+1	11 101 101 10 100 010
INIR	(HL)←(C) B←B−1 HL←HL+1 B=0までくりかえす	11 101 101 10 110 010
IND	(HL)←(C) B←B−1 HL←HL−1	11 101 101 10 101 010
INDR	(HL)←(C) B←B−1 HL←HL−1 B=0までくりかえす	11 101 101 10 111 010

ニーモニック	実行内容	命令コード 76 543 210
OUT (n), A	(n)←A	11 010 011 ← n →
OUT (C), r	(C)←r	11 101 101 01 r 001
OUTI	(C)←(HL) B←B−1 HL←HL+1	11 101 101 10 100 011
OTIR	(C)←(HL) B←B−1 HL←HL+1 B=0までくりかえす	11 101 101 10 110 011
OUTD	(C)←(HL) B←B−1 HL←HL−1	11 101 101 10 101 011
OTDR	(C)←(HL) B←B−1 HL←HL−1 B=0までくりかえす	11 101 101 10 111 011

(注) 前記命令表で使用してある記号の意味は次のとおりです。

r, r'	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

dd, ss	ペアレジスタ
00	BC
01	DE
10	HL
11	SP

qq	ペアレジスタ
00	BC
01	DE
10	HL
11	AF

pp	ペアレジスタ
00	BC
01	DE
10	IX
11	SP

rr	ペアレジスタ
00	BC
01	DE
10	IY
11	SP

b	ビットセット
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

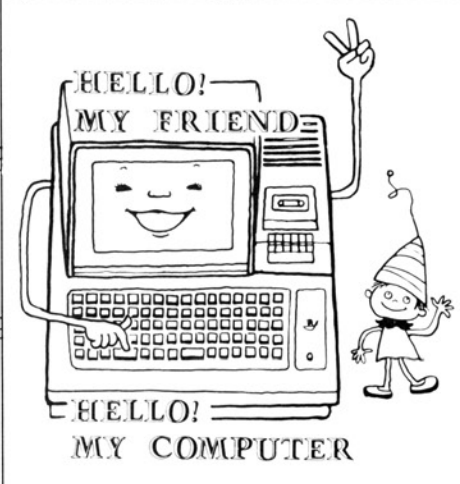
cc	コンディション	
000	NZ	non zero
001	Z	zero
010	NC	non carry
011	C	carry
100	PO	parity odd
101	PE	parity even
110	P	sign positive
111	M	sign negative

t	p
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

∧：論理積
∨：論理和
⊕：排他的論理和

s：r, n, (HL), (IX+d), (IY+d)　　m：r, (HL), (IX+d), (IY+d)
CY：キャリーフリップフロップ　　m_b：ビットb又はロケーションm
$(ペアレジスタ)_H$：ペアレジスタの上位8ビット　　$(ペアレジスタ)_L$：ペアレジスタの下位8ビット

ADC、SUB、SBC、AND、OR、XOR、CPのニーモニックに対する命令コードは、それらの□の内容をADDグループの□の内容と入れかえたものになります。
DECの命令コードはその□の内容をINCグループの□の内容と入れかえたものになります。
ローテイト、シフトグループ及びビットセット、リセット、テストグループにおける命令コードの□についても同様の処理を行うことを意味します。


HELLO!
MY FRIEND
HELLO!
MY COMPUTER

パーソナル・コンピュータ用語集

I／O（input／output）

入出力とも呼ばれるが、インプット（入力）とはコンピュータの中へ外部からデータやプログラムを入れることで、アウトプット(出力)は逆に出すことです。またI/Oポートというのは、データやプログラムを出し入れする際の、物理的な入口と出口の総称です。

アキュムレータ（accumulator）

累算器ともいいます。ＣＰＵ内に組みこまれたレジスタのひとつで、演算結果を一時的に蓄えるメモリーとしての機能のほか、アキュムレータの内容と決められた数値との代数和をとったり、論理演算を行うこともできます。また記憶された数値の桁送りや、補数などの操作も行い重要な役割を持っているレジスターです。

アクセスタイム（access time）

命令やデータをメモリーへ書き込む、あるいはメモリーから読み出すのにかかる時間で、書き込みか読み出しの信号が出されてから、それらを実行し始めるまでの時間をいいます。

アスキー（ASCii）

American standard code for information interchange の略。コンピュータでよく使われる標準コードで、アルファベット、数字、？や＄などの記号１２８種類を７ビット＋１ビットのパリティ符号の計８ビットで表わしたコードです。

アセンブラ（assembler）

０と１の集まりで表わされるコンピュータの命令コードやデータを人間に扱いやすいように、アルファベットや数字に一対一に対応するよう変換した言語をアセンブリ言語と呼びます。アセンブリ言語で構成したプログラムを、０と１の符号に翻訳しなおすプログラムがアセンブラと呼ばれるプログラムです。

アドレス（address）

メモリー内を適当に区分して番号付けをした番地のことです。データや命令をメモリーに書きこみ、読み出したりするときは、アドレスを指定することによって行なわれます。

アナログ（analogue）

ある量を表わすのに連続的な量を用いることで、液体温度計やテスターの針などがその例です。デジタルに対応するものです。

アンダフロー（underflow）

コンピュータ内において、演算結果が小さすぎてコンピュータ内で扱いうる数の範囲に満たない状態のこと。

インタフェース（interface）

ふたつの装置の接点で、情報のやりとりをするための仲人のような働きをします。ふつうコンピュータではＣＰＵと入出力装置の間にはいって、情報のやりとりをスムーズにするための回路やプログラムを指します。

インタープリター（interpreter）

メモリーにたくわえられたプログラム言語を順次取りだしては解釈しながら直接実行してゆくプログラムです。代表例はＢＡＳＩＣで、プログラム言語を一括して翻訳するのではなく、１プログラムごとに同時通訳してゆく特徴をもっています。

インタラプト（interrupt）

インタラプトとは割り込みという意味ですが、コンピュータではひとつのプログラムの実行を中断して、急ぎの処理のプログラムを動かしたのち、また中断したところから再開する一連の動作を指します。

エディタ（editor）

コンピュータでエディタとは編集プログラムのことです。これはコンピュータの中にあらかじめバッファと呼ばれるエリアをとっておき、キーボードや紙テープから入力されたデータやプログラムに対して、必要な修正をほどこして、修正ずみの内容を打ちだすためのプログラムです。

オクタル（octal）

０から７までの数字を使う８進数表示のことです。

ＬＳＩ（Large Scale Integration）

大規模集積回路。１チップ上に１０００個以上の素子が組まれたＩＣを指します。マイクロプロセッサには約６０００個の素子が集積されています。

オーバフロー（overflow）

コンピュータで処理するとき、演算した結果が、コンピュータ内で扱いうる数値の範囲を越えた状態のこと。

オブジェクトプログラム（object program）

目的プログラムとも呼ばれ、コンピュータが実行できる形式に並んだプログラムで、機械語で作られています。ふつうこのプログラムを作りだすには、コンパイラやアセンブラのような翻訳プログラムが必要です。

オフライン（off line）

ある端末機器がCPUと直結していない状態を指します。コンピュータを利用するシステムを離れていえば、Aという機器とBという機器、あるいはAというシステムとBというシステム、それからAという機器とBというシステムとの間で、情報を受け渡すラインが結合されていず、人手を必要とする状態を呼びます。

オペレーティングシステム（operating system）

よくOSと呼ばれていますが、コンピュータ全体をもっとも効率よく使うために、いろいろのプログラム言語と端末機器と利用者との関係を、統一的に管理して無駄なく制御するためのプログラムの集合を指します。つまり利用者が最小限の操作をするだけで、きわめて機能的にシステムを動かせるように作られた制御プログラムを意味することが多いようです。

オンライン（on line）

コンピュータを利用するシステムにおいて、CPUが他の機器を直接制御して働かせる方式をいいます。直接制御の状態とは、端末機器からCPUまでのつながりで、人手を必要としないまま、情報の受け渡しができる状態です。国鉄のみどりの窓口や銀行のキャッシュ・カードを利用した業務はオンラインの代表例です。

キャラクタ（character）

アルファベット、カナ文字、数字、記号など、コンピュータが記憶したり表示したりすることのできる文字をいいます。キャラクタ・ディスプレイ装置と呼ばれるものは、通常キャラクタをコード化してROMに記憶させており、テレビ画面に表示することきは、キャラクタジェネレータと呼ばれる回路を通して、文字が表示される仕組みになっております。

グラフィック　ディスプレイ（graphic display）

コンピュータでテレビ画面に図形を描かせるものです。

コボル（COBOL）

common business oriented language の略。主として事務用のデータ処理を目的として開発されたプログラム言語で、命令語がファイル処理に適するようにつくられているのが特徴です。

コマンド（command）

命令、指令という意味ですが、プログラムによる命令ではなく、特にキーボードなど入力装置から与える命令をコマンドといいます。

コンパイラ（compiler）

機械語に近いアセンブリ言語より、人間の言語に近いプログラム言語を使ってプログラムを組み、それを一括して機械語へ翻訳するためのプログラムです。 FORTRAN、COBOLはコンパイラの代表例といえます。

サブルーチン（subroutine）

プログラムの全体をルーチンと呼びますが、繰り返して行なう部分を1つの小さなプログラムとして扱い、必要な時そのプログラムを何度でも実行できるようにしたものをサブルーチンといいます。 BASIC のプログラムでは GOSUB という命令で指定されてサブルーチンへ移り、RETURNという命令で元のプログラムへ戻ります。プログラムの繰り返しを省き、メモリーを節約することができます。

CPU（中央処理装置）

central processing unit の略。コンピュータの演算・制御機能の心臓部分です。マイクロコンピュータのCPUをマイクロプロセッサと呼ぶこともあります。

シミュレーション（simulation）

ある現象や変化を模型化して、模型がどう反応するかを数学的に表現して分析することです。大きくいえば占いもシミュレーションの一種で、特にコンピュータで実現するとき、コンピュータ・シミュレーションと呼ばれます。物まねと覚えておけばわかりやすいでしょう。

スタック（stack）

スタックとは棚という意味で、あるデータを順々に重ね、取り出すときは最後に重ねたもの、つまり上の方から順々にとるようになっています。コンピュータではサブルーチンからの戻り番地を記憶するために用いたりします。

ストリング（string）

ストリングとは糸、一連、一列という意味があり、コンピュータでは文字の列を指すことが多い。たとえば"フジサン　ハ　タカイ"という文字もコンピュータで処理するときはストリングとして扱われます。

ソーティング（sorting）

与えられたデータをある目的に沿って、特定の順番に並べかえること。たとえば数の大きさの順、アルファベット順、特定の項目の順などに並べかえ分類することです。

ソースプログラム（source program）

オブジェクトプログラム（機械語プログラム）と比較して呼ばれるもので、ふつうは機械語に翻訳する前のプログラムのこと。高水準な言語（たとえば FORTRAN、COBOL、あるいはアセンブラ言語など）で書かれたプログラムを総称し、いったん機械語へ翻訳しなければコンピュータで実行することはできません。

タイムシェアリング（time sharing）

時分割ともいいます。１台のコンピュータを、同時に多くの利用者に使えるようなシステムをさしています。たとえばテレタイプでデータの入力を行なうとき早くても１文字あたりせいぜい0.5秒ほどの時間がかかりますがコンピュータの処理はずっと短時間で済んでしまいます。それで、複数個のターミナルを時間を区切って使い、見かけ上、同時にたくさんのターミナルを利用することができるのです。

ターミナル（terminal）

人間の手によってコンピュータに情報を与えたり、情報をコンピュータから受けとる端末装置のことで、キーボードやカセットテープレコーダやブラウン管ディスプレイ、あるいはテレタイプ、フロッピーディスクなどもターミナルと呼ばれます。

ディスプレイ（display）

ブラウン管の画面に文字や図形を表わす装置です。厳密にはＣＲＴディスプレイ、ＴＶディスプレイ、ブラウン管ディスプレイなどと使います。

デジタル（digital）

ある量を表わすのに数値を用いることで、デジタル時計のように、時刻が数字で表現されます。これに反して、砂時計・日時計のように連続的な量で表わされるのが、アナログと呼ばれます。

データ（data）

コンピュータで処理する目的で、アルファベット、数字などにより表現された集まりを指す。ふつうにはプログラムに対して、プログラムによって処理される数値や文字列のことです。

テレタイプ（teletype）

コンピュータ・ターミナルのひとつで、電信回路を使って情報をキー入力したり、送られて来た情報を文字に直して印字する機能があります。

デバッグ（debbug）

バッグとは「虫」のことで、デバッグとは「虫取り」のこと。転じて、プログラム上でのミスを見つけだし、修正することです。

ニーモニックコード（mnemonic code）

アセンブリ言語の命令コードのこと。０と１の集まりで表わされるマイクロプロセッサの命令符号を、人間にわかりやすいように記号化したものです。

バイト（byte）

８ビットの大きさの情報量を１バイトという単位で表わします。

バイナリ（binary）

２進数のように、０と１とでデータなどを表現する方法。コンピュータ内の信号の取り扱いはすべてバイナリで行なわれています。

バス（bus）

母線とも呼ばれる線路の束です。バスは大別して３つあり、データを送るデータバス、アドレス指定のためのアドレスバス、データをどう処理するかを制御するためのコントロールバスがあります。

バッファ（buffer）

コンピュータ内の各装置をつなぐバスの途中に置かれデータ転送の調整を行うものです。両方の装置の転送速度の違いに応じて、歩調を合せる役目などを持っています。

ＢＣＤ（２進化１０進数）

binary coded decimal number の略。１０進数の各桁の数字を４ビットの２進数で表わして並べた数。１０進数との変換に便利な形ですが、たとえば通常の２進数１６ビットでは１０進数の０から６５５３５が表わされるのに対して、ＢＣＤ１６ビットでは０から９９９９までの１０進数しか表わせないので、大きな数の取り扱いには有利ではありません。

ビット（bit）

ランプが点灯したり消えたり、スイッチが接続されたり開いたりといった電気信号の状態は、あれか－これか、オン－オフ、真－偽の二者択一の状態を表わします。これを情報の単位とすればわかりやすく、世界中で話し合って決められたのがビットです。bit とは binary（２進の）digit（桁）の略で、たとえば８ビットは２進数の８桁で表わされる情報の大きさです。

ＰＲＯＭ（programmable read only memory）

再書き込みのできる読み出し専用メモリーのこと。いったん書き込まれた内容を消すには、紫外線やＸ線の光エネルギーをあてる方法がよく使われます。

ファイル（file）

ある目的のために組織的に集められた情報をいいます。利用法から大別すると、シーケンシャルファイルとランダムファイルがあります。シーケンシャルファイルは、ファイル内の記録が一定の順序で並んでいるもので、磁気テープのファイルが代表的です。また、ランダムファイルはファイル内の記録順序とは関係なく、任意の記録を取り出したり、任意の位置に書き込んだりできるもので、フロッピーディスクが代表的なものです。

フォートラン（FORTRAN）

formula translatorの略。主として科学技術計算用に開発されたプログラム言語で、広範囲の計算式を普通の演算記号に近い形で表現できるのが特徴です。

プリンタ (printer)

決められた紙にコンピュータの処理結果を出力するための装置。大別すると、タイプライタと同じような方式のインパクト型と、化学的・物理的な現象を利用したノン・インパクト型のふたつがあります。前者は鮮明な画面が得られるという利点に対して、音がうるさい、高価であるという欠点を持ちます。後者にはインクを吹きつけるもの、放電現象あるいは熱をあてると色が変わる特殊紙を利用する方法などがあります。この方式だと、価格が比較的安いこと、音が静かであることの利点があるものの、耐久性に難があること、印字があまりきれいでないという欠点もともないます。

フローチャート (flowchart)

コンピュータがどんな手順で実行すればよいかを分析して、だれがみてもわかるような図形でまとめたものです。フローチャートにはプログラムの順序、組立て方などを主にして、全体の仕事内容をあますことなく簡潔に表現されていなければなりません。図形記号の詳細はＪＩＳで決められています。

フロッピーディスク (floppy disk)

ディスケット (diskett) とも呼ばれ、ディスク状のプラスチックに磁性膜がコーティングされたものを指します。これを高速で回転しながら情報の読み書きを行なうもので、容量が大きく、しかも安価なメモリーファイルとしての特徴があります。

ベーシック (BASIC)

ビギナーズ　オールパーパス　シンボリック　インストラクション　コード
Biginner's All-purpose Symbolic Instruction Code の略。マイクロコンピュータで広く用いられているプログラム言語で、簡単で覚えやすく、またコンピュータと会話をするような使い方ができる特徴があります。

ボー (baud)

データや命令などの情報を送るときの速度を表わす単位で１秒間に送られる情報のビット数で示します。たとえば１２００ボーといえば、８ビットのデータが１５０個１秒間に送られることを表わしています。

メモリー (memory)

大量のデータや命令を記憶する装置。マイクロコンピュータの内部記憶装置にはＩＣのＲＯＭとＲＡＭが用いられています。カセットテープやフロッピーディスクなどは外部記憶装置と呼ばれるメモリーです。メモリーに記憶される情報は０と１の２進符号で、記憶される場所はアドレスによって指定されています。

モニター (monitor)

コンピュータの操作を容易にし、いま動かしてみたいシステムを効率よく使うために用意された管理プログラムのことです。

ライトペン (light pen)

ブラウン管ディスプレイに付属して使われるペン型の手動入力装置。先端に光を感知する素子を内蔵しており、先端をブラウン管の蛍光面上の希望位置におくと、コンピュータもその位置を認識するので、文字や図形の修正や編集が容易にできます。

ラム (RAM)

ランダム　アクセス　メモリー
random access memory の略。アドレスのいかんを問わず、データの読み出しや書き込みに必要な時間が同じメモリーを指します。わかりやすくいえば、音楽テープに音楽を記憶させて聞くだけでなく、どんどん新しい音楽に入れかえることができるのです。ただし、電気を切ってしまうと記憶させた内容は消えてしまう性質があります。

ラベル (label)

アセンブリプログラムで、データや命令を識別するために付けられる名前のこと。

リアルタイム (real time)

コンピュータで使うときは、情報が入力されてから結果が出力されるまでの時間が即時的なことを指します。たとえば電話をかけるとき、ダイヤルを回し終わって相手が出るまでの時間のずれは、実用上では即時とみなされます。コンピュータでも入力が終わってから結果が出てくるまでにはやはりいくらかの時間のずれがあり、このおくれが実用上問題にならない限り実時間と呼ぶことができます。

ロム (ROM)

リード　オンリー　メモリー
read only memory の略。これはいったん書き込んでしまえば、再度書き込むことは原則として不可能なメモリーで、ちょうどレコード盤のようなものです。いったん吹き込まれた音楽は、レコード盤がこわれるまで保持され、新しい音楽を受けつけません。ＲＯＭも記憶させた情報は電気が切られても忘れないように作られています。

ローダー (loader)

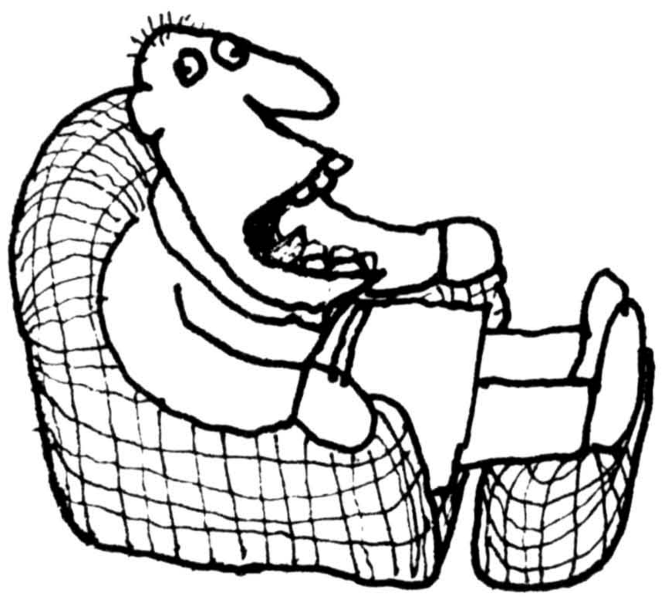
外部記憶装置の情報を内部記憶装置に読み込むためのプログラムでモニタに内蔵されています。カセットテープからのロードをはじめ、紙テープやフロッピーディスクからのロードのとき実行するプログラムです。

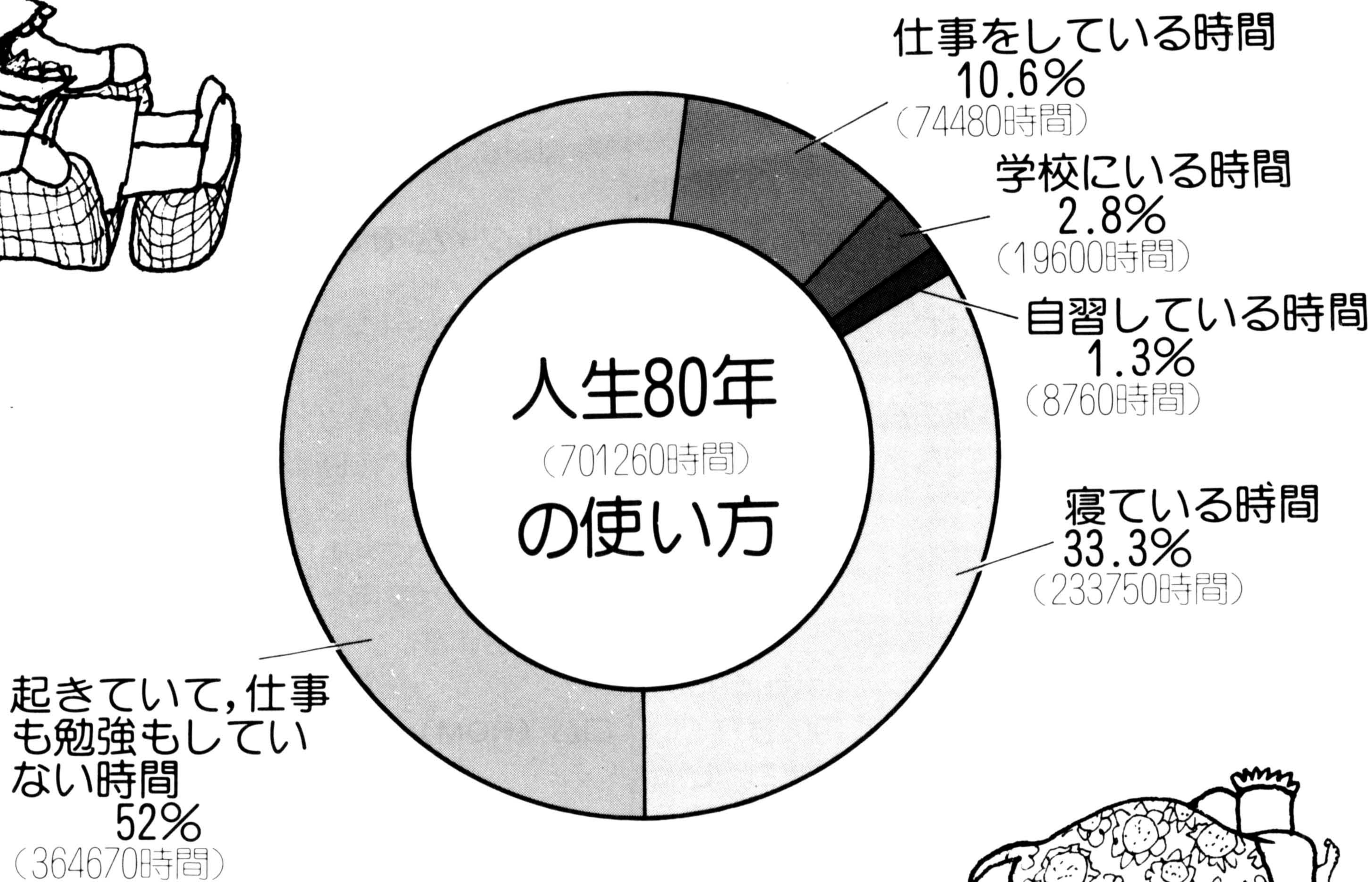
ワード (word)

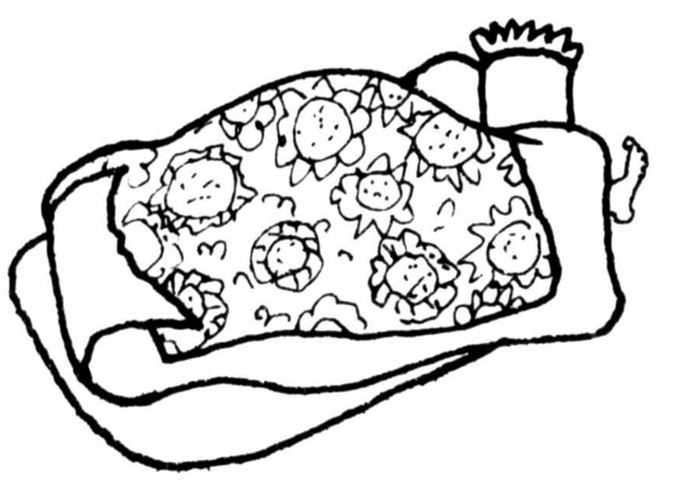
メモリー内のひとつのアドレスに記憶できるデータの単位をワードまたは語といいます。ワードの大きさはコンピュータ内のデータの単位となります。たとえば８ビットコンピュータというときワードの大きさが８ビットであることが示されます。

701,260時間

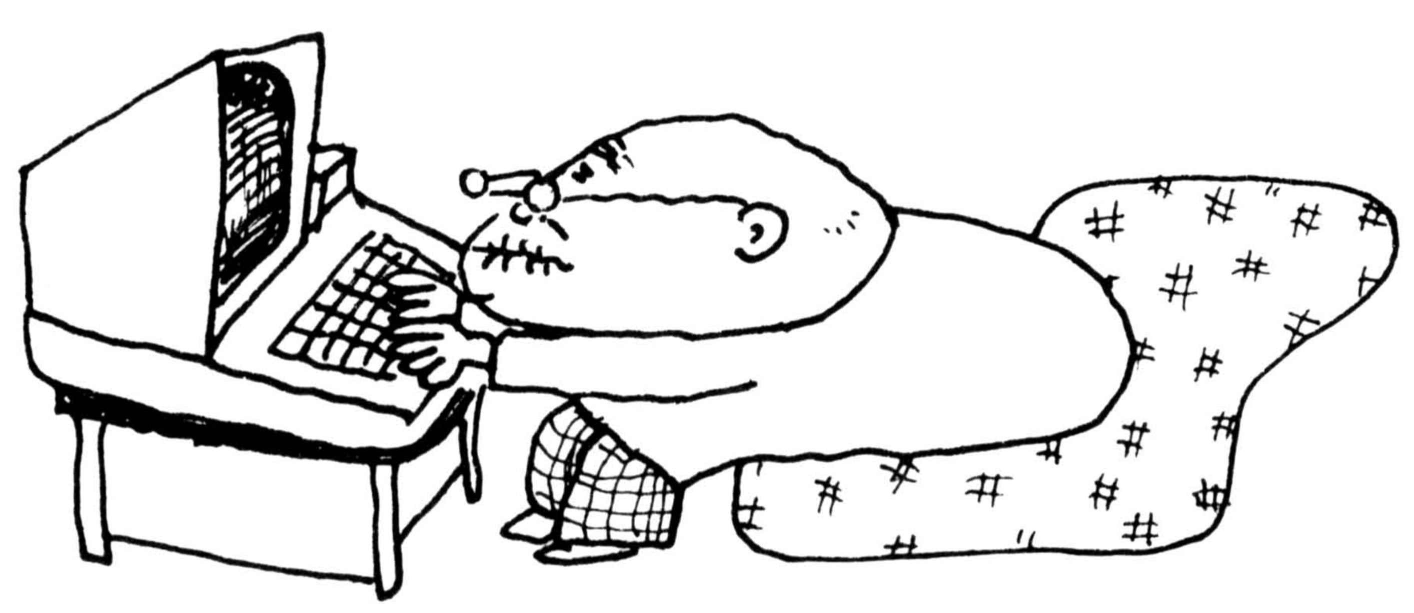
もう、コンピュータをあなたの頭脳の一部ででもあるかのように、自由に使えるようになりましたか。エ？　忙しくてじっくり取り組む時間がなかなか作れない、ですって？……お互いに生きるためには本当にせわしない世の中ですね。
ところで、あなたがどれくらい多忙な一生をすごしているかを、コンピュータを使って予測してみる、というのはいかがでしょうか？ここでは例に、日本人の平均より少し長目に生きるものとし、人生80年として計算してみました。学校は小・中・高・大学と16年間、１年に245日登校し、１日５時間授業を受けます。帰宅後は１日も欠かさず1.5時間ずつ自習する、という真面目青年。そして大学を出てからは、サラリーマンとして毎年245日間８時間勤務を続けるものとします。
１日の睡眠時間は８時間、１年を365.24日として計算してみました。







いかがですか、人生80年の間に勉強をしている時間は合計4.1％、仕事をしている時間は10.6%。さて、あなたご自身のこれからの時間の使い方、もう少しこまかく分析し予測してみたらいかがでしょう。そのときは、コンピュータを相手にする時間もぜひ入れておいてくださいね。



ＢＡＳＩＣ ＳＰ－５０３０仕様のまとめ

ＭＺ－８０シリーズＢＡＳＩＣ　ＳＰ－５０３０の仕様をまとめて示します。

１．形式

中間言語形式インタープリタ、大きさ：約１４Ｋバイト、先頭アドレス：＄１２００番地

２．数値データ

浮動小数点形式、有効数字８桁×$10^{\pm 19}$の実数

実数範囲：０．２７１０５０５５Ｅ－１９～０．９２２３３７２０Ｅ＋１９

１６進数：ＬＩＭＩＴ、ＰＯＫＥ、ＰＥＥＫ、ＵＳＲの各コマンドにおいて直接１６進番地を指定する場合にのみ使用可能であり、１６進４桁の前に"＄"マークをつけて示します。

（例）　ＬＩＭＩＴ　＄８ＦＦＦ、　ＵＳＲ（＄Ａ０００、Ｘ＄）

３．数値変数

数値変数の種類：数値変数、１次元配列数値変数、２次元配列数値変数

（１）　数値変数……………………変数名は、はじめの２文字が有効です。また、はじめの文字は英文字を使わなくてはなりません。特殊文字、あるいは、ＢＡＳＩＣで使用しているキーワードまたは、キーワードを含む名前を使うこともできません。

（例）　Ａ、Ｘ、Ｆ１、ＡＡ　などは可
ＡＢＣとＡＢＤは同じ変数とみなします
ＤＡＴＡ、ＸＤＡＴＡ、Ａ＃などは不可

（２）　１次元配列数値変数……配列の最大は２５５までで、ＤＩＭによって配列を定義しなくてはなりません。（１）の数値変数と同様の条件で変数名が決められます。配列はカッコ内に添字を置いて示します。

（例）　ＤＩＭ　Ｑ（１００）　によってＱ（０）からＱ（１００）まで定義されます。

（３）　２次元配列数値変数……配列の添字を２つもつ変数で、各添字について定義できる配列の最大は２５５までですが、メモリの使用状態によって制限されることになります。

（例）　ＤＩＭ　Ａ１（７、７）によってＡ１（０、０）からＡ１（７、７）まで８×８＝６４の配列要素をもつ２次元配列Ａ１（　、　）が定義されます。

４．ストリングデータ

ストリングの最大長：２５５文字

ストリングで扱う内部コード：各文字のＡＳＣｉｉコード列、終端にキャリッジリターンコード０ＤＨがつきます。

５．ストリング変数

ストリング変数の種類：ストリング変数、１次元配列ストリング変数、２次元配列ストリング変数

ストリング変数の形式：ストリング変数は、数値変数の場合と同様の条件で選んだ名前に＄記号をつけることによって表現します。
変数名の規則、配列型ストリング変数の定義などは、数値変数の場合と同様

（例）　Ａ＄、ＳＴ＄、ＮＩ＄　は通常のストリング変数
ＮＡＭＥ１＄とＮＡＭＥ２＄は同じストリング変数とみなします
ＴＩ＄、ＣＨＲ＄（　）などは特殊ストリング変数
ＤＩＭ　Ｓ＄(３、３)　によって、Ｓ＄(０、０)からＳ＄(３、３)まで４×４＝１６の配列要素をもつ２次元ストリング配列Ｓ＄（　、　）が定義されます。

６．その他の仕様、定数など

（1） 文番号：１～６５５３５の間で使用できます。

（2） LOAD、VERIFY、SAVEにおけるファイルネームの有効桁数：最大１６文字

（3） CURSOR　X、Yコマンドにおけるの XとY：X＝０～３９、Y＝０～２４

（4） SET　X、Y　または　RESET　X、YにおけるXとY：X＝０～７９、Y＝０～４９

（5） TI$に代入できるストリング：6桁の１０進数のストリングであり、最初の2文字は"００"から"２３"まで（時間を表わす）、次の2文字は"００"から"５９"まで(分を表わす)最後の2文字も"００"から"５９"まで（秒を表わす）となっています。

（6） FOR……NEXTループのネスティング：最大１５まで重ねることが可能

（7） GOSUB　によるネスティング：最大１５まで重ねることが可能

（8） DEF　FNによる関数定義における関数定義のネスティング：最大6まで重ねることが可能

（9） INP、OUT命令による入出力ポートアドレス：０～２３９（１０進数）
入出力されるデータ：8ビットデータ、１０進数で０～２５５

（10） オートリピート機能：カーソルコントロールキー［CURSOR（↑↓）］［CURSOR（←→）］［CLR HOME］、スペースキー［SP］、挿入・削除キー［INST DEL］の各キーについてオートリピート機能があります。

（11） 円周率πの値：π＝３．１４１５９２７

MZ-80

SERIES


Hyperboreer
Hippomolgen
Kikonen
Skylla
Trinakrie
Kyklopen
Ithaka
Sparta
Odygia
Kreta
Paphlagonier
Lydien
Phönicien
Cypern
Theben
Liby
Region des Tages
Erember
Athiopen
Pygmäen
Fluss
Okeanos

MZ-80
SERIES


Hyperboreer
Hippomolgen
Kikonen
Skylla
Trinakrie
Kyklopen
Itaka
Sparta
Kreta
Lydien
Paphlagonier
Theben
Liby
Region des Tages
Pygmaen
Phönicien
Erember
Athiopen
Fluss
Okeanos

シャープ株式会社

本　　社	〒545 大阪市阿倍野区長池町22-22 電話(06)621－1221(大代表)
産業機器事業本部	〒639-11 奈良県大和郡山市美濃庄町492 電話(07435)3－5521(大代表)

お客様ご相談窓口

北海道	(011)642-4649	東　北	(0222)96-4649	関　越	(0286)35-1151
東　京	(03)893-4649	北　陸	(0762)49-4649	中　部	(052)322-4649
近　畿	(06)643-4649	中　国	(082)874-4649	四　国	(0878)33-4649
九　州	(092)572-4649	沖　縄	(0988)62-2231		

シャープ株式会社　国内家電営業本部 システム機器営業企画部

〒545 大阪市阿倍野区長池町22番22号　電話(06)621－1221(大代表)

シャープエンジニアリング

㈱北海道エンジニアリング	〒063 札幌市西区24軒1条7丁目3-17号	電話(011)641－4649
㈱東　北エンジニアリング	〒983 仙台市萩野町2丁目8番地9	電話(0222)96－4649
㈱関　越エンジニアリング	〒320 宇都宮市不動前4丁目2番41号	電話(0286)35－1151
㈱東　京エンジニアリング	〒114 東京都北区東田端2丁目13番17号	電話 (03)800－1221
㈱北　陸エンジニアリング	〒921 石川県石川郡野々市町字御経塚1096番地の1	電話(0762)49－4649
㈱中　部エンジニアリング	〒454 名古屋市中川区山王3丁目5番5号	電話(052)322－4649
㈱近　畿エンジニアリング	〒556 大阪市浪速区恵美須西1丁目2番9号	電話 (06)643－4649
㈱中　国エンジニアリング	〒731-01 広島市安佐南区祇園町大字西原2249の1	電話(082)874－2281
㈱四　国エンジニアリング	〒760 高松市木太町1861番地の3	電話(0878)33－4649
㈱九　州エンジニアリング	〒816 福岡市博多区井相田2丁目12番地の1	電話(092)572－4650
沖　縄シャープ電機㈱	〒900 沖縄県那覇市曙2丁目10番地1	電話(0988)62－2231

シャープビジネス株式会社

〒545 大阪市阿倍野区長池町22-22　電話(06)621－1221(大代表)

販推部	〒162 東京都新宿区市谷八幡町8番地	電話 (03)260－1161(大代表)
札幌支店	〒063 札幌市西区24軒1条7丁目3番17号	電話(011)641－3631
仙台支店	〒983 宮城県仙台市萩野町2丁目8番地の9	電話(0222)94－3247
東京支店	〒130 東京都墨田区石原2丁目12番3号	電話 (03)625－5111
千葉支店	〒280 千葉市南町1丁目5番20号	電話(0472)63－4043
横浜支店	〒235 横浜市磯子区中原1丁目2番23号	電話(045)751－3215
埼玉支店	〒330 埼玉県大宮市宮原町2丁目107番地2号	電話(0486)63－5159
宇都宮支店	〒320 宇都宮市不動前4丁目2番41号	電話(0286)37－3576
新潟支店	〒950 新潟市上所中1丁目7番21号	電話(0252)83－1795
名古屋支店	〒454 名古屋市中川区山王3丁目5番5号	電話(052)332－2631(代表)
金沢支店	〒921 石川県石川郡野々市町御経塚1096番1号	電話(0762)49－1240
大阪支店	〒556 大阪市浪速区恵美須西1丁目2番9号	電話 (06)643－3021(代表)
京都支店	〒601 京都市南区上鳥羽菅田町48番地	電話(075)681－4361(代表)
神戸支店	〒658 神戸市東灘区魚崎北町1丁目6番地18号	電話(078)431－4361(代表)
広島支店	〒731-01 広島市安佐南区祇園町大字西原2249番地の1	電話(082)874－4925
高松支店	〒760 香川県高松市木太町1861番地の3	電話(0878)33－4255
福岡支店	〒816 福岡市博多区井相田2丁目12番地の1	電話(092)572－2611
沖縄支店	〒900 沖縄県那覇市曙2丁目10番地の1	電話(0988)61－7360(代表)

シャープ株式会社